Universidad Juárez Autónoma de Tabasco
División Académica de Ciencias Básicas

SELECCIÓN DE VARIABLES EN MODELOS DE REGRESIÓN LINEAL CONTROLANDO LA TASA DE FALSOS POSITIVOS

Tesis

Que para obtener el grado de:

MAESTRO EN CIENCIAS MATEMÁTICAS

Presenta

LIC. LEONARDO ALFONSO MARTÍNEZ GONZÁLEZ.

Directores de tesis:

DRA. ADDY MARGARITA BOLÍVAR CIMÉ (UJAT).

Dr. ROGELIO RAMOS QUIROGA.

Cunduacán, Tabasco. Octubre 2024

Declaración de Autoría y Originalidad

En la Ciudad de Cunduacán, Tabasco, el día 19 del mes de agosto del año 2024, el que suscribe Leonardo Alfonso Martínez González alumno del Programa de Maestría en Ciencias Matemáticas con número de matrícula 192A21003, adscrito a la División Académica de Ciencias Básicas, de la Universidad Juárez Autónoma de Tabasco, como autor de la tesis presentada para la obtención del título Maestro en Ciencias Matemáticas y titulada Selección de Variables en Modelos de Regresión Lineal Controlando la Tasa de Falsos Positivos dirigida por la Dra. Addy Margarita Bolívar Cimé y el Dr. Rogelio Ramos Quiroga.

DECLARO QUE:

La Tesis es una obra original que no infringe los derechos de propiedad intelectual ni los derechos de propiedad industrial u otros, de acuerdo con el ordenamiento jurídico vigente, en particular, la LEY FEDERAL DEL DERECHO DE AUTOR (Decreto por el que se reforma y adiciona diversas disposiciones de la Ley federal del Derecho de Autor del 01 de julio de 2020 regularizando y aclarando y armonizando las disposiciones legales vigentes sobre la materia), en particular, las disposiciones referidas al derecho de cita.

Del mismo modo, asumo frente a la Universidad cualquier responsabilidad que pudiera derivarse de la autoría o falta de originalidad o contenido de la Tesis presentada de conformidad con el ordenamiento jurídico vigente.

Villahermosa, Tabasco a 19 de agosto de 2024.

Leonardo Alfonso Martínez González

DIRECCIÓN

06 de agosto de 2024

LIC. LEONARDO ALFONSO MARTÍNEZ GONZÁLEZ
PASANTE DE LA MAESTRÍA EN CIENCIAS MATEMÁTICAS
PRESENTE

Por medio del presente y de la manera más atenta, me dirijo a Usted para hacer de su conocimiento que proceda a la impresión del trabajo titulado "**SELECCIÓN DE VARIABLES EN MODELOS DE REGRESIÓN LINEAL CONTROLANDO LA TASA DE FALSOS POSITIVOS**", en virtud de que reúne los requisitos para el EXAMEN PROFESIONAL y obtener el grado de Maestro en Ciencias Matemáticas.

Sin más por el momento, reciba un cordial saludo.

ATENTAMENTE

DIVISIÓN ACADÉMICA DE CIENCIAS BÁSICAS

DRA. HERMICENDA PÉREZ VIDAL
DIRECTORA

C.c.p.- Archivo

DIR´DRA.HPV/JP´DRA.EAM/jkal

Carta de Cesión de Derechos

Cunduacán, Tabasco a 19 de agosto de 2024

Por medio de la presente manifiesto haber colaborado como AUTOR(A) y/o AUTORES(RAS) en la producción, creación y/o realización de la obra denominada Selección de Variables en Modelos de Regresión Lineal Controlando la Tasa de Falsos Positivos.

Con fundamento en el artículo 83 de la Ley Federal del Derecho de Autor y toda vez que, la creación y/o realización de la obra antes mencionada se realizó bajo comisión de la Universidad Juárez Autónoma de Tabasco; entendemos y aceptamos el alcance del artículo en mención, de que tenemos el derecho como autores de la obra, y la Universidad Juárez Autónoma de Tabasco mantendrá en un 100% la titularidad de los derechos patrimoniales por un periodo de 20 años sobre la obra en la que colaboramos, por lo anterior, cedemos el derecho patrimonial exclusivo en favor de la Universidad.

COLABORADORES

Lic. Leonardo Alfonso Martínez González
EGRESADO

Dra. Addy Margarita Bolívar Cimé
DIRECTORA

Dr. Rogelio Ramos Quiroga
CODIRECTOR

TESTIGOS

Dr. Aroldo Pérez Pérez

Dr. Edilberto Nájera Rangel

Agradecimientos

Al concluir este importante proyecto de investigación, no puedo sino expresar mi más profundo agradecimiento a todas aquellas personas e instituciones que, de una u otra manera, han sido fundamentales en este proceso.

Quiero agradecer a mi padre, cuya guía y apoyo incondicional me han dado la fortaleza necesaria para seguir adelante, su confianza ha sido un pilar en cada etapa de mi formación y ha sido la motivación constante para alcanzar esta meta. A mis hermanos, quienes han sido mis compañeros de vida, les agradezco por su paciencia, apoyo moral y por estar siempre presentes, brindándome palabras de aliento en los momentos más difíciles. La unión y el respaldo de mi familia han sido esenciales para superar los desafíos que se presentaron a lo largo de este camino.

Un agradecimiento especial va dirigido a mis dos asesores, quienes han desempeñado un papel crucial en mi desarrollo académico. Agradezco a mi asesora la Dra. Addy Margarita Bolívar Cimé, no solo por su invaluable orientación y sabiduría durante este proyecto, sino también por los años de convivencia y apoyo continuo, su compromiso con mi formación ha sido una fuente de inspiración, y su disposición para guiarme más allá de las exigencias académicas ha sido un verdadero regalo, sin su esfuerzo y dedicación, este trabajo no habría alcanzado la calidad y profundidad que hoy presenta. A mi asesor, Dr. Rogelio Ramos Quiroga, le extiendo mi gratitud por su respaldo constante y por las perspectivas valiosas que aportó a esta investigación, su tiempo y dedicación han sido invaluables, y su influencia se refleja claramente en los resultados obtenidos.

Finalmente, deseo expresar mi agradecimiento al Consejo Nacional de Humanidades, Ciencia y Tecnologías, cuya beca fue un apoyo significativo durante el desarrollo de este proyecto. Más allá de la ayuda financiera, agradezco la confianza depositada en mí, la cual fue un impulso para mantenerme enfocado y comprometido con la excelencia.

A todos ellos, les extiendo mi más sincero agradecimiento, sabiendo que este logro es tanto mío como de ustedes.

Índice general

Índice de Tablas

Índice de figuras

Introducción

En muchos campos de la ciencia, comúnmente observamos una variable de respuesta junto con un gran número de variables explicativas potenciales, y deseamos ser capaces de descubrir cuáles son las variables que están verdaderamente asociadas con la respuesta. El control de la tasa de falsos positivos en el análisis de datos y la reducción de la tasa de falsos positivos en investigación científica, son aspectos importantes en la validación de resultados y la toma de decisiones basadas en datos.

Algunos métodos para reducir la tasa de falsos positivos en investigación científica incluyen el uso de pruebas de significancia estadística apropiadas, el ajuste de p-valores para múltiples pruebas, el uso de técnicas de validación cruzada y la replicación de estudios.

En el análisis de datos, es importante utilizar métodos como la validación cruzada, la selección de características y la regularización para controlar la tasa de falsos positivos.

Supongamos que tenemos registros de una variable de respuesta de interés Y y muchas variables explicativas potenciales X_i en n unidades de observación. Nuestras observaciones obedecen un modelo de regresión lineal clásico

$$Y = X\beta + \epsilon,$$

donde $Y \in \mathbb{R}^n$ es un vector de respuestas, $X \in \mathbb{R}^{(n\times p)}$ es una matriz conocida, $\beta \in \mathbb{R}^p$ es un vector de coeficientes desconocido, y $\epsilon \sim N(0, \sigma^2 I)$ es un vector aleatorio de ruido.

Una técnica común para predecir el valor de datos desconocidos mediante el uso de otro valor relacionado y conocido es la regresión lineal. La regresión

lineal es un método estadístico que modela la relación entre una variable dependiente y una o más variables independientes, y se utiliza para pronosticar el valor de la variable dependiente basándose en los valores de las variables independientes.

Entre los métodos más utilizados para seleccionar variables en un modelo de regresión lineal clásico tenemos el Stepwise y Lasso, los cuales trabajan minimizando el error de tipo I, es decir de la probabilidad de rechazar una variable cuando se debió aceptar.

El modelo Stepwise es un método de selección de variables en el que se van añadiendo o eliminando variables del modelo de forma secuencial, con el objetivo de encontrar el conjunto óptimo de variables predictoras, por otro lado, Lasso es un método de regularización que penaliza los coeficientes de las variables menos importantes, forzándolos a ser cero y así seleccionando automáticamente las variables más relevantes.

En ese sentido podemos seleccionar variables con modelos que se concentren en reducir el error de tipo II, es decir, la probabilidad de aceptar una variable cuando se debió rechazar. En [Barber and Candès, 2015] se expone la metodología de Knockoff, la cual consiste en un método para reducir el error tipo II en la selección de variables en modelos de regresión lineal.

El propósito de esta tesis es mostrar el funcionamiento de esta metodología de selección de variables vía knockoff en modelos de regresión lineal en los casos $p < n$, $p < n < 2p$ y $n \ll p$ (dimensión p mucho más grande al tamaño de la muestra n). También se analizará su tasa de falsos positivos en comparación a otros métodos y se mostrará una aplicación a datos reales.

La tesis está dividida de la siguiente manera: en el capítulo 1 se da una breve explicación del modelo de regresión lineal y algunos métodos de selección de variables; en el capítulo 2 se presenta el concepto de descubrimientos falsos en pruebas de hipótesis múltiples, así como algunas pruebas en donde se intenta minimizar dicho error; en el capítulo 3 se expone la metodología de knockoff, así como su aplicación; en el capítulo 4 se presentan los resultados de un estudio por simulación del comportamiento de los métodos de selección

Stepwise, Lasso y knockoff, también una aplicación a datos reales; finalmente se presentan las conclusiones de la tesis y dos apéndices, uno con detalles técnicos y otro con los códigos en R utilizados en este trabajo.

Capítulo 1

Modelos lineales y selección de variables

1.1. Introducción

En este capítulo se presentarán conceptos básicos del Modelo Lineal y de algunos métodos de selección de variables, que serán empleados más adelante en el análisis estadístico de datos.

A continuación, presentamos definiciones y resultados de acuerdo a [Draper and Smith, 1998].

1.2. Modelo de Regresión Lineal general

El modelo de regresión lineal general en estadística es un enfoque para modelar la relación entre una variable dependiente y una o más variables independientes. Se utiliza para predecir el valor de la variable dependiente en función de las variables independientes. El modelo asume que la relación entre las variables es lineal y se puede expresar mediante una ecuación matemática.

En este modelo, se ajusta una línea recta a los datos para minimizar la distancia entre los valores observados y los valores predichos por el modelo. La ecuación del modelo de regresión lineal general es

$$Y = \beta X + \epsilon,$$

donde Y es un vector de observaciones de $(n \times 1)$, X es una matriz de características conocidas de $(n \times p)$, β es un vector de parámetros de $(p \times 1)$, ϵ es

un vector de errores de $(n \times 1)$, con $E[\epsilon] = 0$ y $Var[\epsilon] = I\sigma^2$, por lo que los elementos de ϵ no están correlacionados.

Ya que $E[\epsilon] = 0$, una manera alternativa de escribir el modelo es

$$E[Y] = X\beta.$$

La suma de los cuadrados de los errores está dada por

$$\begin{aligned} \epsilon'\epsilon &= (Y - X\beta)'(Y - X\beta) \\ &= Y'Y - \beta'X'Y - Y'X\beta + \beta'X'X\beta \\ &= Y'Y - 2\beta'X'Y + \beta'X'X\beta. \end{aligned} \tag{1.1}$$

La estimación de mínimos cuadrados de β es el vector b, que cuando se sustituye en β minimiza $\epsilon'\epsilon$. Esto puede ser determinado por la diferenciación de (1.1) con respecto a β y estableciendo la ecuación matricial resultante al igualar a cero, al mismo tiempo que se reemplaza β por b. Esto proporciona la *ecuación normal*

$$(X'X)b = X'Y. \tag{1.2}$$

Surgen dos casos principales: ya sea que (1.2) consiste en p ecuaciones independientes en p incógnitas, o algunas ecuaciones dependen de otras de modo que haya menos de p ecuaciones independientes en las p incógnitas. Si algunas de las ecuaciones normales depende de otras, $X'X$ es singular, así que $(X'X)^{-1}$ no existe, entonces el modelo debería ser expresado en términos de menos parámetros o restricciones adicionales en los parámetros deben ser impuestas o asumidas; si las p ecuaciones normales son independientes, $X'X$ es no singular y su inversa existe, en este caso la solución de las ecuaciones normales puede ser escrita como

$$b = (X'X)^{-1}X'Y.$$

Esta solución b tiene las siguientes propiedades:

1. Es un estimador de β que minimiza el error de la suma de cuadrados $\epsilon'\epsilon$, independientemente de cualquier distribución de los errores.
   Una suposición de que los errores ϵ se distribuyen normalmente no es necesaria para obtener la estimación b, pero esto se requiere más tarde

para hacer pruebas que dependen de la suposición de normalidad, como la prueba t o F, o para obtener intervalos de confianza basados en estas distribuciones.

2. Los elementos de b son funciones lineales de las observaciones $Y_1, Y_2, \ldots, Y_n$ y proporcionan estimaciones insesgadas de los elementos de β, las cuales tienen la mínima varianza (entre cualquier función lineal de las $Y_i's$ que proporciona estimaciones insesgadas), independientemente de las propiedades de la distribución de los errores.
   Supongamos que tenemos una expresión $T = l_1Y_1 + l_2Y_2 + \cdots + l_nY_n$, la cual es una función lineal de las observaciones $Y_1, Y_2, \ldots, Y_n$, que es usada como un estimador de un parámetro θ. Entonces T es una variable aleatoria cuya distribución de probabilidad dependerá de la distribución de las $Y_i's$. Si repetidamente tomamos muestras de $Y_i's$ y obtenemos las correspondientes $T's$, generamos la distribución empírica de T. Si hacemos esto o no, la distribución de T tendrá un valor medio definido que escribiremos como $E[T]$ y una varianza que escribiremos como $V[T]$. Si sucede que la media de la distribución de T es igual al parámetro θ que estimamos con T, es decir, $E[T] = \theta$, entonces decimos que T es un estimador insesgado de θ. La palabra estimador es normalmente usada cuando nos referimos a la expresión teórica para T en términos de muestras $Y_i's$. Un valor numérico específico de T puede ser llamada una estimación insesgada para θ. Si tenemos todas las funciones lineales posibles $T_1, T_2, \ldots$ de n observaciones $Y_1, Y_2, \ldots, Y_n$ y si las $T's$ satisfacen

$$\theta = E[T_1] = E[T_2] = \cdots,$$

   es decir, todas son estimadores insesgados de θ, entonces el que tenga el valor más pequeño de $V[T_i]$, $i = 1, 2, \ldots$, es el estimador insesgado de varianza mínima de θ.

1.2.1. Una justificación para usar mínimos cuadrados

Si los errores son independientes y $\epsilon_i \sim N(0, \sigma^2)$, entonces b es el estimador de máxima verosimilitud de β. En términos vectoriales, podemos expresar que $\epsilon \sim N(0, I\sigma^2)$. Esto significa que ϵ sigue una distribución normal multivariada n-dimensional con $E[\epsilon] = 0$ y $V[\epsilon] = I\sigma^2$, es decir, ϵ tiene una matriz de varianza y covarianza cuyos elementos en la diagonal, $V[\epsilon_i]$, $i = 1, 2, \ldots, n$, son

todos σ^2, mientras que los elementos fuera de la diagonal (covarianza entre ϵ_i y ϵ_j con $i \neq j = 1, 2, \ldots, n$) son todos ceros.

La función de verosimilitud para la muestra $Y_1, Y_2, \ldots, Y_n$ es definida en este caso como el producto

$$\prod_{i=1}^{n} \frac{1}{\sigma(2\pi)^{1/2}} e^{-\epsilon_i^2/(2\sigma)^2} = \frac{1}{\sigma^n (2\pi)^{n/2}} e^{-\epsilon'\epsilon/(2\sigma)^2}.$$

Así, para un valor fijo de σ, maximizar la función de verosimilitud es equivalente a minimizar la cantidad $\epsilon'\epsilon$. Note que este hecho puede ser usado para proveer una justificación para el procedimiento de mínimos cuadrados (para minimizar la suma de cuadrados de errores), porque en algunas situaciones el supuesto de que los errores estén distribuidos normalmente es bastante sensible. En cualquier caso, se debe averiguar si este supuesto parece ser violado, examinando los residuales del análisis de regresión.

Si hay conocimiento disponible previo sobre la distribución de los errores, quizá por consideración teórica o por conocimiento previo sólido del proceso de estudio, el argumento de máxima verosimilitud podría ser usado para obtener estimaciones basadas en un criterio distinto de mínimos cuadrados. Por ejemplo, supongamos que los errores ϵ_i, $i = 1, 2, \ldots, n$, son independientes y siguen la distribución doble exponencial

$$f(\epsilon_i) = (2\sigma)^{-1} e^{-|\epsilon_i|/\sigma} \ (-\infty \leqslant \epsilon_i \leqslant \infty)$$

en lugar de la distribución normal

$$f(\epsilon_i) = \frac{1}{\sigma(2\pi)^{1/2}} e^{-\epsilon_i^2/2\sigma^2},$$

como es usualmente asumido. La función de densidad doble exponencial tiene un pico de altura $1/2\sigma$ en $\epsilon_i = 0$, y sus colas tienden a cero cuando ϵ_i tiende a más y menos infinito. Entonces la aplicación del principio de máxima verosimilitud para la estimación de β, asumiendo a σ fija, implicaría la minimización de

$$\sum_{i=1}^{n} |\epsilon_i|,$$

la suma de los errores absolutos, y no la minimización de

$$\sum_{i=1}^{n} \epsilon_i^2,$$

la suma de los cuadrados de los errores.

1.3. Propiedades de los mínimos cuadrados

Asumiendo que $E[\epsilon] = 0$, $Var[\epsilon] = I\sigma^2$, podemos proceder con los siguientes pasos independientemente si los errores están distribuidos normalmente o no:

1. Los valores ajustados son obtenidos de $\widehat{Y} = Xb$.

2. Los vectores de los residuales están dados por $\epsilon = Y - \widehat{Y}$. Se cumple que $\sum_{i=1}^{n} \epsilon_i \widehat{Y}_i = 0$, cualquiera que sea el modelo lineal. Esto se puede ver multiplicando la j-ésima ecuación normal por la j-ésima coordenada de b y sumando los resultados. Si existe un término β_0 en el modelo, también se cumple que $\sum_{i=1}^{n} \epsilon_i = 0$. Los ϵ_i y $\widehat{Y}_i$, $i = 1, 2, \ldots, n$, son los i-ésimos elementos de los vectores ϵ y $\widehat{Y}$, respectivamente. Así $\epsilon'\widehat{Y} = 0 = \widehat{Y}'\epsilon$, $\epsilon' 1 = 0 = 1'\epsilon$ siempre y cuando el modelo contiene β_0.

3. $V[b] = (X'X)^{-1}\sigma^2$ proporciona las varianzas y covarianzas de las estimaciones.

4. Supongamos que X_0' es un vector específico de $(1 \times p)$ cuyos elementos corresponden a una fila de X tal que $\widehat{Y}_0 = X_0'b = b'X_0$ es el valor ajustado en una ubicación especificada definida por X_0. Por ejemplo, si el modelo fuera $Y = \beta_0 + \beta_1 X + \beta_{11} X^2 + \epsilon$, entonces $X_0' = (1, X_0, X_0^2)$ para un valor dado X_0. Entonces $\widehat{Y}_0$ es el valor predicho en X_0 por la ecuación de regresión y tiene varianza

$$V(\widehat{Y}_0) = X_0' V(b) X_0 = X_0'(X'X)^{-1} X_0 \sigma^2.$$

5. Un análisis básico de la tabla de varianza puede ser construido como sigue:

   5a. Si un término β_0 está en el modelo, podemos subdividir la suma de cuadrados en

$$SS(b_0) = \frac{(\sum Y_i)^2}{n} = n\overline{Y}^2,$$

Fuente	df	SS	MS
Regresión	p	$b'X'Y$	$MS_{Regresion}$
Residual	$n-p$	$Y'Y - b'X'Y$	$MS_{Residual}$
Total	n	$Y'Y$	

$$SS(Regresion|b_0) = SS(Reg|b_0) = b'X'Y - \frac{(\sum Y_i)^2}{n}.$$

Estas sumas de cuadrados se basan en 1 y $p-1$ grados de libertad, respectivamente.

5b. Si hay observaciones disponibles repetidas podemos separar la SS residual en SS(error puro) con n_e grados de libertad, la cual estima $n_e\sigma^2$ y SS(falta de ajuste) con $(n-p-n_e)$ grados de libertad. Las repeticiones ahora deben ser repeticiones en todas las coordenadas $X_1, X_2, \ldots, X_k$ de las variables predictoras. Esto proporciona una tabla del análisis de varianza como sigue

Fuente	df	SS	MS	
b_0	1	$SS(b_0)$		
Regresión$\|b_0$	$p-1$	$SS(Reg\|b_0)$	$MS(Reg\|b_0)$	$\}MS_{Regresion}$
Falta de ajuste	$n-p-n_e$	$SS(lof)$	$MS(lof)$	
Error puro	n_e	$SS(pe)$	$MS(pe)$	$\}MS_{Residual}$
Total	n	$Y'Y$		

El estadístico R^2

La razón

$$R^2 = \frac{SS(Reg|b_0)}{Y'Y - SS(b_0)} = \frac{\sum(\widehat{Y}_i - \overline{Y})^2}{\sum(Y_i - \overline{Y})^2} \tag{1.3}$$

es una extensión de la cantidad definida por la recta de la regresión lineal y es el cuadrado del coeficiente de regresión múltiple. Otro nombre para R^2 es el *coeficiente de determinación múltiple.*

R^2 es el cuadrado de correlación entre Y y $\widehat{Y}$ y $0 \leq R^2 \leq 1$. Si el error puro existe, es imposible que R^2 en realidad alcance 1. Un ajuste perfecto para los datos en el cual $\widehat{Y}_i = Y_i$ es un evento improbable en la práctica, que produciría $R^2 = 1$.

Si $\widehat{Y}_i = \overline{Y}_i$, esto es, si $b_1 = b_2 = \ldots = b_{p-1} = 0$, entonces $R^2 = 0$. Así R^2 es una medida de la utilidad de los términos, además de β_0, en el modelo.

El estadístico R^2 ajustada

Supongamos que p es el número total de parámetros en el modelo ajustado (incluyendo a β_0) y SS_{n-p} es la suma de cuadrados de los residuales correspondientes. Tenemos definido el estadístico R^2, una medida de cantidad de variación alrededor de la media de los valores ajustados, como

$$R^2 = \frac{b'X'Y - n\overline{Y}^2}{Y'Y - n\overline{Y}^2} = 1 - \frac{RSS_{n-p}}{CTSS}, \tag{1.4}$$

donde $CTSS$ denota la suma total corregida de los cuadrados $Y'Y - n\overline{Y}^2$ y donde n es el número total de observaciones.
Un estadístico relacionado, el cual es preferido por algunos autores, es la R^2 *ajustada* la cual se define, en nuestro contexto, como

$$R_a^2 = 1 - \frac{(RSS_{n-p})/(n-p)}{(CTSS)/(n-1)} = 1 - (1 - R^2)\left(\frac{n-1}{n-p}\right).$$

En este estadístico se realiza un ajuste a los correspondientes grados de libertad de las cantidades RSS_{n-p} y $CTSS$, la idea es que el estadístico R_a^2 puede ser usado para comparar los valores ajustados no solo de un conjunto específico de datos, sino también de dos o más conjuntos de datos completamente diferentes. La equivalencia de los numeradores en las ecuaciones (1.3) y (1.4) puede establecerse de la siguiente manera:

$$\sum(\widehat{Y} - \overline{Y})^2 = \sum \widehat{Y}_i^2 - (\sum Y_i)^2/n$$

y

$$\begin{aligned} \sum \widehat{Y}_i^2 = \widehat{Y}'\widehat{Y} &= (Xb)'(Xb) \\ &= b'X'Xb \\ &= b'X'Y, \end{aligned} \tag{1.5}$$

ya que $X'Xb = X'Y$, por la ecuación normal.

1.4. Propiedades de los mínimos cuadrados cuando $\epsilon \sim N(0, I\sigma^2)$

La descomposición en el análisis de varianza es solo una igualdad algebraica y no depende de las propiedades distributivas de lo errores. Sin embargo, si

asumimos adicionalmente que $\epsilon_i \sim N(0, \sigma^2)$ y que los ϵ_i son independientes, podemos hacer lo siguiente:

1. Probar la falta de ajuste utilizando la razón

$$\left[\frac{SS(lof)/(n-p-n_e)}{SS(pe)/n_e}\right]$$

como una variable con distribución $F[n-p-n_e, n_e]$ y comparar este valor con su cuantil de cola superior α, $F[(n-p-n_e), n_e, 1-\alpha]$. Si no hay falta de ajuste, $SS(residual)/(n-p) = MS_E$, usualmente llamado s^2, es un estimador insesgado de σ^2. Si la falta de ajuste no puede ser probada, el uso de s^2 como un estimador de σ^2 implica una suposición de que el modelo es correcto. (Si el modelo no es correcto, s^2 generalmente será demasiado grande ya que es una variable aleatoria con una media mayor que σ^2. Sin embargo, debido a la fluctuación de muestreo - ya que es una variable aleatoria - también podría ser demasiado pequeño).

2. Probar la ecuación de regresión general usando la razón cuadrática media

$$\frac{[SS(Reg|b_0)/(p-1)]}{s^2} \tag{1.6}$$

como una variable $F(p-1, v)$, donde $v = n-p$.

La distribución de R^2

Vemos que

$$\begin{aligned} R^2 &= \frac{SS(Regresion|b_0)}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2} \\ &= \frac{SS(Regresion|b_0)}{SS(Regresion|b_0) + ResidualSS} \\ &= \frac{v_1 F}{v_1 F + v_2}, \end{aligned} \tag{1.7}$$

donde la igualdad

$$F = \frac{SS(Regresion|b_0)/v_1}{ResidualSS/v_2}$$

es nuestro estadístico F habitual para probar la regresión general dada b_0, esto es, para probar la hipótesis nula H_0 : que todos los β's (excluyendo β_0)

son cero, contra la hipótesis alternativa H_1 : que al menos una de las β's (excluyendo β_0) no es cero. El valor de β_0 es irrelevante para la prueba. En la ecuación (1.6) podemos establecer $v_1 = p - 1$, $v_2 = n - p$. Bajo H_0, F es distribuida como una variable $F(v_1, v_2)$. Un teorema estadístico nos dice que R^2 sigue una distribución $\beta(\frac{1}{2}v_1, \frac{1}{2}v_2)$, llamada la distribución beta y en este caso con $\frac{1}{2}v_1$ y $\frac{1}{2}v_2$ grados de libertad. Podría probarse H_0 vs H_1 usando la distribución de R^2, el resultado sería exactamente equivalente a la prueba estándar F, el punto crítico para R^2 puede ser obtenida de la ecuación (1.7) con $F(p-1, n-p, 1-\alpha)$ sustituido por F.

Continuando con la lista de propiedades tenemos lo siguiente:

3. Si usamos un estimador s_v^2 para σ^2, los límites de confianza del $100(1-\alpha)\,\%$ para el verdadero valor de la media de Y en X_0 son obtenidos por

$$\widehat{Y}_0 \pm t(v, 1-\frac{1}{2}\alpha) s_v \sqrt{X_0'(X'X)^{-1}X_0}.$$

4. Se tiene que

$$b \sim N(\beta, (X'X)^{-1}\sigma^2).$$

5. Se obtienen intervalos de confianza del $100(1-\alpha)\,\%$ individuales para los diversos parámetros por separado, con la fórmula

$$b_i \pm t(v, 1-\alpha/2) se(b_i), \tag{1.8}$$

donde el "$se(b_i)$" es la raíz cuadrada del i-ésimo término de la diagonal de la matriz $(X'X)^{-1}s^2$. Estos intervalos pueden ser usados para definir un bloque rectangular en el espacio de las β's. Este bloque no es propiamente una región de confianza conjunta para los β's.

6. Se obtiene una región de confianza conjunta del $100(1-\alpha)\,\%$ para todo los parámetros de β de la ecuación

$$(\beta - b)'X'X(\beta - b) = ps^2 F(p, v, 1-\alpha),$$

donde $F(p, v, 1-\alpha)$ es el punto $1-\alpha$ (cuantil de cola superior α) de la distribución $F(p, v)$ y donde s^2 tiene el mismo significado que arriba y el modelo es asumido como correcto. Esta igualdad es la ecuación del contorno de una " forma elíptica "(o más generalmente "forma elipsoidal") en un espacio que tiene tantas dimensiones p, como parámetros en β.

1.4.1. Límites de Bonferroni

Un conjunto más conservador de intervalos (es decir, más amplios) de la forma de (1.8) es obtenida si reemplazamos $t(v, 1-\alpha/2)$ por $t(v, 1-\alpha/(2p))$. Se puede demostrar que estos intervalos tienen conjuntamente un coeficiente de confianza de al menos $1-\alpha$.

1.5. Métodos de Selección de Variables

En muchas ocasiones se tiene un conjunto grande de posibles variables regresoras, una pregunta es saber si todas las variables deben de entrar en el modelo de regresión o bien qué variables deben de entrar o no al modelo. En ocasiones, ajustamos un modelo de regresión teniendo una idea clara de las variables que debemos incluir como regresoras. Es más frecuente, sin embargo, el caso en que sólo tenemos una idea aproximada de la forma adecuada para nuestro modelo, y debemos decidir con criterio estadístico qué regresores deben ser incluidos. Para enfrentar este tipo de situaciones necesitamos, por una parte, criterios de bondad de ajuste, capaces de permitirnos comparar distintos modelos ajustados a una misma muestra. Además, requerimos estrategias de selección de variables que generen de forma automática o semi-automática subconjuntos a partir de todas las posibles configuraciones de modelos, con el objetivo de identificar la opción óptima.

1.5.1. Regresión Stepwise

La regresión paso a paso (stepwise) es la construcción iterativa paso a paso de un modelo de regresión que implica la selección automática de variables independientes. Las definiciones y resultados que se presentan a continuación fueron obtenidos en [Draper and Smith, 1998].

El procedimiento de regresión Stepwise comienza seleccionando una ecuación que incluya la mejor variable X_j, la cual es la más significativa para el modelo según el criterio utilizado. Luego, mediante adiciones sucesivas de variables X_i, una a la vez, siempre que la adición sea significativa, se construye un modelo final en el que todas las variables son significativas. El orden de la adición se determina utilizando los valores estimados de la prueba F para decidir qué

variable debe ingresar a continuación. Se compara el valor estimado F_ϵ más alto con un valor de entrada F_E (seleccionado o predeterminado). Después de agregar una variable, se examina la ecuación para determinar si se debe eliminar alguna.

El procedimiento básico es el siguiente:

Primero, seleccionamos la variable X_i con la mayor correlación con Y (digamos X_1). Así obtenemos la ecuación de regresión lineal de primer orden $\widehat{Y} = f(X_1)$ y verificamos si esta variable es significativa según el criterio utilizado. Si no lo es, la eliminamos y adoptamos el modelo $Y = \overline{Y}$ como el mejor. En caso contrario, buscamos la segunda variable predictiva para incluirla en la regresión.

Examinamos los valores F parciales de las variables que no están en la regresión. La variable X_i con el valor F parcial más grande (supongamos que es X_2) se selecciona, y se ajusta una segunda ecuación de regresión $Y = f(X_1, X_2)$. Se verifica la significancia de la regresión global, se revisa si mejora el valor de R^2, y se examinan los valores F parciales de ambas variables en la ecuación. El valor más bajo de estos valores F parciales se compara con un punto porcentual apropiado de la distribución F. La variable predictora correspondiente se retiene o elimina de la ecuación según la significancia de la prueba. Estas pruebas sobre el "predictor actual menos útil en la ecuación" se llevan a cabo en cada etapa del procedimiento Stepwise.

Un predictor que podría haber sido el mejor candidato en una etapa inicial puede volverse innecesario en una etapa posterior debido a las relaciones con otras variables actuales en la regresión. Para verificar esto, se utiliza el criterio del valor F parcial para cada variable en la regresión en cada etapa del cálculo. Luego, se compara el valor más bajo de estos (que puede estar asociado con la variable más reciente o con una anterior) con un punto porcentual preseleccionado de la distribución F apropiada o un valor F predeterminado.

Ejemplo 1.5.1 *Para ilustrar el funcionamiento de la metodología de Stepwise en la selección de variables, se realizó una regresión lineal por simulación con* $n = 200$ *observaciones de* $p = 100$ *variables, de tal forma que solo* 30 *de estas variables fueran significativas, es decir, que en la regresión su correspondiente*

β_i es distinta de cero. Los subíndices de las variables con coeficientes distintos de cero fueron:

4	10	15	19	20	24	31	36	37	40
42	44	58	61	65	66	67	68	70	75
76	79	80	85	87	93	94	97	99	100.

Utilizando la metodología de Stepwise, y el paquete "bigstep" de R, se seleccionaron las siguientes variables, en azul se marcan las correctas.

15	19	36	37	40	44	64	66	67	68
70	75	76	79	81	86	87	93	94	97
99	100.								

Podemos notar que el método de Stepwise seleccionó varias variables de forma correcta y pocas incorrectas.

Ejemplo 1.5.2 *La metodología de Stepwise se puede extender al caso de dimensión alta ($p > n$). Para ilustrar este caso, se realizó una regresión lineal por simulación con $n = 100$ observaciones de $p = 300$ variables, de tal forma que solo 30 de estas variables fueran significativas, es decir, que en la regresión su correspondiente β_i es distinta de cero. Los subíndices de las variables con coeficientes distintos de cero fueron:*

5	22	26	42	53	67	85	103	117	137
153	171	172	182	187	189	191	208	220	222
224	232	238	242	243	255	266	275	276	289.

Utilizando la metodología de Stepwise, y el paquete "bigstep" de R, se seleccionaron las siguientes variables, en azul se marcan las correctas.

16	22	25	26	41	67	85	103	142	147
154	171	172	188	207	221	222	229	231	232
236	238	241	253	255	265	266	275	281	282
285	287	297.							

Podemos notar que el método de Stepwise en este caso de dimensiona alta, seleccionó pocas variables de forma correcta y varias incorrectas.

1.5.2. Método de mínimos cuadrados penalizados

Como se menciona en [González et al., 2015], la idea clave del método que se describe a continuación es la penalización, la cual evita el sobre ajuste debido al gran número de variables predictoras imponiendo una penalización o término de penalización, el cual obliga a que algunas componentes del vector de parámetros $\beta = (\beta_1, \ldots, \beta_p)$ sean cero. La elección del parámetro de penalización es fundamental, por lo cual es necesario un procedimiento que estime el valor de dicho parámetro a partir de los datos.

Para ello se propone un enfoque unificado a través de mínimos cuadrados penalizados, que consiste en estimar el vector de parámetros β minimizando la siguiente expresión:

$$\sum_{i=1}^{n}(y_i - x_i'\beta)^2 + \lambda \sum_{j=1}^{p} p_\lambda(|\beta_j|), \tag{1.9}$$

donde p_λ es la función de penalización que será diferente para cada método y λ es el parámetro de penalización, el cual debe ser elegido a través de algún procedimiento basado en los datos muestrales. De este modo estimamos el vector β como el que minimiza la expresión (1.9) y se denotará por $\widehat{\beta}_n$. Notemos que si $\lambda = 0$, este estimador es igual al estimador de mínimos cuadrados ordinarios.

Se proponen tres condiciones deseables que un método de penalización debería cumplir:

- Esparsidad: efectuar selección de variables automáticamente, es decir, tener la potencialidad de fijar variables a cero.
- Continuidad: ser continuo en los datos para evitar inestabilidad en la predicción.
- Insesgadez: tener bajo sesgo, especialmente para valores grandes de los coeficientes β_j.

A continuación se describen dos métodos de regresión penalizada, el método Ridge y Lasso, los cuales difieren en los tipos de penalización $p(\lambda)$ utilizada.

1.5.3. Método Ridge

Las definiciones y resultados que se presentan a continuación fueron obtenidos en [Hastie et al., 2015].

La regresión de Ridge reduce los coeficientes de regresión al imponer una penalización en su tamaño. Los coeficientes de Ridge minimizan una suma residual de cuadrados penalizada,

$$\widehat{\beta}^{ridge} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}. \tag{1.10}$$

Aquí $\lambda \geq 0$ es un parámetro de complejidad que controla la cantidad de contracción, cuanto mayor es el valor de λ, mayor es la cantidad de contracción, los coeficientes se reducen hacia cero (y entre sí).

Una forma equivalente de escribir el problema de Ridge es

$$\widehat{\beta}^{ridge} = \arg\min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2, \text{sujeto a } \sum_{j=1}^{p} \beta_j^2 \leq t, \tag{1.11}$$

que hace explícita la restricción de tamaño en los parámetros. Existe una correspondencia uno a uno entre el parámetro λ en (1.10) y t en (1.11). Cuando hay muchas variables correlacionadas en un modelo de regresión lineal, sus coeficientes pueden llegar a estar mal determinados y exhibir una varianza alta. Un coeficiente positivo grande en una variable puede ser cancelado por un coeficiente negativo igualmente grande de la variable correlacionada. Al imponer una restricción de tamaño en los coeficientes, como en (1.11) este problema se resuelve. Las soluciones de Ridge no son equivalentes bajo la escala de las entradas, por lo que normalmente se normalizan las entradas antes de resolver (1.10). Además, notemos que el intercepto β_0 ha quedado fuera del término de penalización.

La penalización del intercepto haría que el procedimiento dependa del origen elegido para Y; es decir, agregar una constante c a cada uno de los objetivos y_i no solo resultaría en una desviación de las predicciones por la misma cantidad c. Se puede mostrar que la solución a (1.10) puede ser separada en dos

partes, después de la reparametrización usando entradas centradas, cada x_{ij} es reemplazada por $x_{ij} - \overline{x}_j$. Estimamos β_0 por $\overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$. Los coeficientes restantes se estiman mediante una regresión de Ridge sin intercepto, usando las x_{ij} centrales. En adelante, suponemos que este centrado se ha realizado, por lo que la matriz de entradas X tiene p (en lugar de $p+1$) columnas. Escribiremos el criterio en (1.10) en forma matricial

$$RSS(\lambda) = (y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta, \tag{1.12}$$

la solución de la regresión Ridge está dada por

$$\widehat{\beta}^{ridge} = (X^T X + \lambda I)^{-1} X^T y, \tag{1.13}$$

donde I es la matriz identidad de $p \times p$. Note que con la elección de penalización cuadrática $\beta^T\beta$, la solución de la regresión Ridge es nuevamente una función lineal de y. La solución agrega una constante positiva a la diagonal de $X^T X$ antes de obtener la inversa. Esto hace que el problema sea no singular, incluso si $X^T X$ no es de rango completo, y fue la principal motivación para la regresión de Ridge cuando se introdujo por primera vez por [Hoerl and Kennard, 1970]. La descripción tradicional de la regresión Ridge comienzan definiendo (1.13).

1.5.4. Método Lasso

Las definiciones y resultados que se presentan a continuación fueron obtenidas en [Friedman et al., 2001].

Dada una colección de N pares de respuestas predictivas $\{(x_i, y_i)\}_{i=1}^{N}$, Lasso encuentra la solución $(\hat{\beta}_0, \hat{\beta})$, para el problema de optimización

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\} \tag{1.14}$$

$$\text{sujeto a} \quad \sum_{j=1}^{p} |\beta_j| \leq t.$$

La restricción $\sum_{i=1}^{n} |\beta_j| \leq t$ puede ser escrita de manera más compacta como la restricción $\| \beta \|_1 \leq t$. Además (1.14) es usualmente representada usando

la notación matriz-vector. Sea $y = (y_1, \ldots, y_N)$ que denota el N-vector de respuesta y sea X una matriz de $N \times p$ con $x_i \in \mathbb{R}^p$ en su i-ésima fila, entonces el problema de optimización (1.14) puede ser reescrito como:

$$\underset{\hat{\beta}_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \parallel y - \beta_0 \mathbf{1} - X\beta \parallel_2^2 \right\} \tag{1.15}$$

$$\text{sujeto a} \quad \parallel \beta \parallel_1 \leq t,$$

donde $\mathbf{1}$ es el vector de N unos y $\| \cdot \|_2$ denota la norma euclidiana usual sobre vectores. La cota t limita la suma de los valores absolutos de la estimación de los parámetros. Dado que una estimación de parámetros reducidos corresponde a un modelo más fuertemente restringido, esta cota limita qué tan bien podemos ajustar los datos.

Por lo general, primero estandarizamos los predictores X para que cada columna esté centrada ($\frac{1}{N}\sum_{i=1}^N x_{ij} = 0$) y tenga varianza unitaria ($\frac{1}{N}\sum_{i=1}^N x_{ij}^2 = 1$). Sin estandarización, las soluciones de Lasso dependerían de las unidades (por ejemplo pies contra metros) utilizadas para medir los predictores. Por otra parte, normalmente no estandarizaríamos si las características se midieran en las mismas unidades. Por conveniencia, también asumimos que los valores de salida y_i se han centrado ($\frac{1}{N}\sum_{i=1}^N y_i = 0$). Estas condiciones de centrado son convenientes, ya que significan que podemos omitir el término de intersección β_0 en la optimización del Lasso. Dada una solución óptima de Lasso $\hat{\beta}$ sobre los datos centrados, podemos recuperar las soluciones óptimas para los datos no centrados, en este caso, $\hat{\beta}$ es igual, y la intersección $\hat{\beta}_0$ viene dada por:

$$\hat{\beta}_0 = \overline{y} - \sum_{j=1}^{p} \overline{x}_j \hat{\beta}_j,$$

donde $\overline{y}$ y $\{\overline{x}_j\}_1^p$ son las medias originales. Por esta razón omitimos la intersección β_0 para Lasso. A menudo es conveniente reescribir el problema del Lasso en la llamada forma lagrangiana

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2N} \parallel y - X\beta \parallel_2^2 + \lambda \parallel \beta \parallel_1 \right\}, \tag{1.16}$$

para algún $\lambda > 0$. Por la dualidad lagrangiana, hay una correspondencia uno a uno entre el problema restringido (1.14) y la forma lagrangiana (1.16). Para cada valor de t en el rango donde la restricción $\| \beta \|_1 \leq t$ esta activa, existe un valor correspondiente de λ que produce la misma solución de la forma lagrangiana (1.16). Por el contrario, la solución $\hat{\beta}_\lambda$ al problema (1.16) resuelve el problema acotado con $t = \| \hat{\beta}_\lambda \|_1$.

En algunas descripciones del método de Lasso, el factor $1/2N$ que está en (1.14) y (1.16) es reemplazado por $1/2$ o simplemente por 1, aunque esto no hace diferencia en (1.14) y corresponde a una simple reparametrización de λ en (1.16). Este tipo de estandarización hace que los valores de λ sean comparables para diferentes tamaños de muestra (útil para la validación cruzada).

La teoría del análisis convexo nos dice que las condiciones necesarias y suficientes para una solución al problema (1.16) toma la forma

$$-\frac{1}{N}\langle x_j, y - X\beta \rangle + \lambda s_j = 0, \quad j = 1, 2, \ldots, p. \tag{1.17}$$

Aquí cada s_j es una cantidad desconocida igual al $sign(\beta_j)$ si $\beta_j \neq 0$ y algún valor en $[-1, 1]$. Este es un subgradiente para la función valor absoluto. En otras palabras, las soluciones $\hat{\beta}$ al problema (1.16) son las mismas que las soluciones $(\hat{\beta}, \hat{s})$ en (1.17). Este sistema es una forma de las llamadas condiciones de Karush–Kuhn–Tucker (KKT) para el problema (1.16).

Ejemplo 1.5.3 *Para ilustrar el funcionamiento de la metodología de la regresión Lasso en la selección de variables, utilizaremos la regresión simulada del ejemplo 1.5.1.*

Para la elección del parámetro λ, una opción simple es utilizar validación cruzada. Se eligen un conjunto de valores para λ y se calcula el error de validación para cada valor, se elige el valor de λ para el cual el error ha sido menor y finalmente se reajusta el modelo con todas las observaciones disponibles con el valor de λ escogido. Así escogemos el valor $\lambda = 0.06742703$.

Entonces con la metodología de la regresión Lasso, y el paquete "glmnet" de R, se seleccionaron las siguientes variables, donde en azul se marcan las

correctas.

4	5	10	12	15	17	18	19	20	22
24	25	27	28	30	31	32	36	37	38
39	40	41	42	44	55	58	60	61	62
65	66	67	68	70	75	76	77	79	80
81	85	86	87	90	92	93	94	96	97
98	99	100.							

Podemos notar que la regresión Lasso, en este ejemplo, selecciona todas las variables que sí son distintas de cero, pero también selecciona varias que no lo son.

Ejemplo 1.5.4 *La regresión Lasso se puede extender al caso de dimensión alta ($p > n$). Para ilustrar este caso, utilizaremos la regresión simulada del ejemplo 1.5.2. La elección del parámetro λ se realiza de la misma manera que en el ejemplo 1.5.3, así el valor es $\lambda = 2.721152$. Los subíndices de las variables significativas son:*

5	6	22	25	26	32	40	42	48	53
64	67	72	75	85	87	103	116	117	118
121	137	142	153	159	170	171	172	174	178
182	187	189	190	191	192	195	208	209	214
220	221	222	224	229	232	236	238	241	242
243	255	266	269	271	275	276	282	289	294.

Podemos notar que la regresión Lasso en este ejemplo de dimensión alta seleccionó a todas las variables que son distintas de cero, pero también varias que no lo son.

Capítulo 2

Control de Falsos Positivos

2.1. Introducción

En muchos campos de la ciencia, comúnmente observamos una variable de respuesta junto con un gran número de variables explicativas potenciales, y deseamos ser capaces de descubrir cuáles son las variables que están verdaderamente asociadas con la respuesta. Al mismo tiempo, necesitamos saber que la *tasa de falsos positivos* (*false discovery rate*), la fracción esperada de los falsos positivos entre todas las variables seleccionadas, no es demasiado alta, para asegurarle al científico que la mayoría de las variables seleccionadas son verdaderas y replicables.

En este capítulo se mostrará la teoría del control de falsos positivos. Las definiciones y resultados que se presentan fueron obtenidas de [Efron, 2012], [Simes, 1986], [Holm, 1979], [Hommel, 1988], [Hochberg, 1988] y [Benjamini and Hochberg, 1995].

2.2. Prueba de hipótesis múltiples

Como se menciona en [Casella and Berger, 2002], una hipótesis es un enunciado acerca de un parámetro poblacional. El objetivo de una prueba de hipótesis es decidir, con base en una muestra de la población, cuál de dos hipótesis complementarias es verdadera.
En general, una prueba de hipótesis estadística posee la siguiente forma:

$$H_0 : \theta \in \Theta_0 \quad \text{vs} \quad H_1 : \theta \in \Theta_0^c,$$

donde θ denota un parámetro poblacional, H_0 es la hipótesis nula, H_1 es la

alternativa, Θ_0 es un subconjunto del espacio parametral Θ, y Θ_0^c es el complemento de Θ_0.

Definiremos una prueba de hipótesis múltiple como una prueba en la que intervienen $N > 1$ hipótesis nulas $H_{01}, H_{02}, \ldots, H_{0N}$, donde el interés radica en realizar pruebas estadísticas simultáneas para determinar cuántas y cuáles de ellas han de ser rechazadas.

Es importante realizar pruebas simultáneas ya que al realizar $N > 1$ pruebas individuales por separado no se controla el error de tipo I. Por ejemplo, supongamos que tenemos

$$H_{01} : \theta_1 = 0 \quad \text{vs} \quad H_{02} : \theta_2 = 0,$$

y estadísticos z_1 y z_2 tales que, rechazamos H_{01} si $z_1 > c_1$ y rechazamos H_{01} si $z_2 > c_2$.

Ahora bien, sea $\alpha \in (0, 1)$ la probabilidad de error de tipo I y

$$P(\text{rechazar } H_{01} | H_{01} \text{ es verdadera}) = \alpha,$$

$$P(\text{rechazar } H_{02} | H_{02} \text{ es verdadera}) = \alpha.$$

Por otra parte, para la prueba múltiple se tiene que

$$\begin{aligned} &P(\text{equivocarse al menos una vez}) = 1 - P(\text{no equivocarse}) \\ &= 1 - P(\text{no equivocarse en } H_{01}) P(\text{no equivocarse en } H_{02}) \\ &= 1 - (1 - \alpha)(1 - \alpha) = 1 - (1 - \alpha)^2. \end{aligned}$$

Así, si consideramos $\alpha = 0.05$, el error de tipo I de la prueba múltiple sería

$$P(\text{equivocarse al menos una vez}) = 1 - (1 - 0.05)^2 \approx 0.1.$$

De manera informal, podemos decir que los procedimientos de pruebas de hipótesis múltiples (PHM) intentan controlar el número de resultados incorrectamente significativos, mientras se prueban simultáneamente un gran

número de hipótesis estadísticas.

Para realizar lo anterior, es fundamental definir lo que se entiende por una adecuada tasa de error, que es básicamente una manera de conceptualizar los errores que se cometen al realizar en paralelo un gran número de contrastes de hipótesis.

Consideraremos los siguientes tipos de errores:

Definición 2.2.1 ***La tasa de error familiar (FWER)*** *es la probabilidad de rechazar al menos un H_i verdadero, es decir, de cometer al menos un error de tipo I.*

Definición 2.2.2 ***La tasa de descubrimientos falsos (FDR)*** *es la proporción esperada de errores tipo I entre las hipótesis rechazadas.*

2.2.1. Descubrimientos verdaderos y falsos.

Queremos probar N hipótesis nulas $H_{01}, H_{02}, \ldots, H_{0N}$ con base en un conjunto de datos X y tomando en cuenta alguna regla de decisión D, la cual produce una decisión de "nulo" o "no nulo" para cada uno de los N casos, es decir, D acepta o rechaza $H_{0i}, i = 1, 2, \ldots, N$, basándose en X.

		Decisión		
		nulo	no nulo	
Realidad	nulo	$N_0 - a$	a	N_0
	no nulo	$N_1 - b$	b	N_1
		$N - R$	R	N

Tabla 2.1: Descubrimientos verdaderos y falsos.

Una regla de decisión D a rechazado R de N hipótesis nulas, ver tabla 2.1, donde a de estas decisiones fueron incorrectas, es decir, fueron *"descubrimientos falsos"*, mientras que b de ellos fueron *"descubrimientos verdaderos"*. La proporción de descubrimientos falsos (Fdp) es igual a a/R.
La tabla 2.1 presenta un resultado hipotético del rendimiento de D, donde N_0 de los N casos eran en realidad nulos, de los cuales D designa a no nulos (incorrectamente) y $N_0 - a$ nulos (correctamente). Similarmente, de los N_1

que eran en realidad no nulos, b son designados por D como no nulos (correctamente) y $N_1 - b$ como nulos (incorrectamente). De los $R = a + b$ rechazos totales, a fueron descubrimientos falsos y b descubrimientos verdaderos.

La tasa de error familiar (FWER), de acuerdo a la tabla, es igual a $P(a > 0)$. En la situación de prueba clásica de un único caso, $N = 1$, se tiene que N_0 o N_1 es igual a 1 y el otro 0. Entonces

$$P(a = 1|N_0 = 1) = \alpha,$$

la cual es la tasa de error de tipo I, o tamaño, de la regla de decisión y

$$P(b = 1|N_1 = 1) = \beta,$$

la cual es la potencia de la regla.
La tasa de descubrimientos falsos (FDR) sería igual a

$$FDR = E[Fdp] = E\left[\frac{a}{R}\right].$$

Ejemplo 2.2.3 *Consideremos al método de Stepwise y la regresión Lasso como esta regla de decisión D, aplicada a la selección de variables de las regresiones simuladas en los ejemplos 1.5.1 y 1.5.2.*

Tenemos para la regresión del ejemplo 1.5.1

$$\begin{aligned} FDR_{Stepwise} &= \frac{3}{22} = 0.1363636, \\ FDR_{Lasso} &= \frac{23}{53} = 0.43396226. \end{aligned} \tag{2.1}$$

Y para la regresión del ejemplo 1.5.2

$$\begin{aligned} FDR_{Stepwise} &= \frac{20}{33} = 0.6060606, \\ FDR_{Lasso} &= \frac{30}{60} = 0.5. \end{aligned} \tag{2.2}$$

2.3. Métodos de pruebas de hipótesis múltiples

Esta metodología surge cuando se quiere contrastar simultáneamente varias hipótesis, dando una protección adicional a la probabilidad de rechazar hipótesis nulas verdaderas.

De manera general, la forma estándar de realizar pruebas de hipótesis múltiples consiste en una extensión natural del caso de una prueba de hipótesis simple. Un algoritmo estándar para pruebas de hipótesis múltiples es el siguiente:

Para $i = 1, \ldots, N$, donde N es el número de hipótesis de prueba, elegimos estadísticos de prueba $T_1, T_2, \ldots, T_N$ con correspondientes p-valores $p_1, p_2, \ldots, p_N$ para las pruebas de hipótesis $H_1, H_2, \ldots, H_N$ y aplicamos un procedimiento de prueba de hipótesis múltiple para determinar cuáles hipótesis se han de rechazar.

2.3.1. Pruebas múltiples de Bonferroni

El procedimiento clásico de **pruebas múltiples de Bonferroni** es usualmente llevado a cabo para un conjunto $H_0 = \{H_1, H_2, \ldots, H_N\}$, en el cual la hipótesis específica $H_i \in H_0$ es rechazada si $p_i \leq \alpha/N$, para $i = 1, 2, \ldots, N$.

La desigualdad de Boole

$$P\left(\bigcup_{i=1}^{N}(p_i \leq \alpha/N)\right) \leq \sum_{i=1}^{N} P(p_i \leq \alpha/N) \leq N\alpha/N = \alpha \quad (0 \leq \alpha \leq 1),$$

asegura que la probabilidad de rechazar al menos una hipótesis nula cuando todas son verdaderas no es más grande que α.

Así

$$FWER = P\left(\bigcup_{i=1}^{N}(p_i \leq \alpha/N)\right) \leq \alpha \quad (0 \leq \alpha \leq 1),$$

es decir, Bonferroni controla la tasa de error familiar.

Aunque el procedimiento de Bonferroni es relativamente simple, resulta ser considerablemente conservador, ya que, si N es suficientemente grande, la cota α/N se vuelve muy pequeña, de manera que, sólo aquellas pruebas con evidencia en contra excepcionalmente evidente serían detectadas.

Como consecuencia, exigir control a un nivel muy pequeño de α, especialmente cuando N es considerablemente grande, viene con el costo de cometer un gran número de errores de tipo II, es decir, no rechazar hipótesis que en realidad son falsas.

Ejemplo 2.3.1 *Utilizando la regresión lineal simulada en el ejemplo 1.5.1, podemos calcular el $p-$valor asociado a la prueba $H_0 : \beta_i = 0$ contra $H_1 : \beta_i \neq 0$, $i = 1, \ldots, p$. Los $p-$valores obtenidos son los siguientes, de color azul están los $p-$valores correspondientes a las betas distintas de cero y consideramos $\alpha = 0.05$, entonces $\alpha/100 = 5 * 10^{-4}$.*

$5.940486e-01$	$1.167430e-01$	$2.614159e-01$	$4.091238e-69$	$9.155224e-02$
$3.927445e-01$	$7.860042e-01$	$8.254403e-01$	$9.887991e-01$	$2.650218e-68$
$5.143174e-01$	$2.827543e-02$	$7.970381e-01$	$3.399692e-01$	$2.579670e-67$
$1.002627e-01$	$6.360909e-02$	$5.589601e-04$	$1.183116e-65$	$3.924809e-72$
$1.758180e-01$	$3.066264e-01$	$5.825061e-01$	$5.091490e-66$	$5.593056e-02$
$5.310642e-01$	$9.041794e-01$	$6.493548e-02$	$8.276563e-01$	$3.050559e-01$
$3.244553e-70$	$4.668017e-01$	$3.811319e-01$	$3.502565e-01$	$3.410515e-01$
$8.790030e-69$	$3.773595e-68$	$3.276987e-02$	$4.477669e-01$	$1.881122e-69$
$9.464218e-03$	$3.681254e-64$	$1.782607e-01$	$1.226016e-68$	$6.927141e-01$
$4.477647e-01$	$9.245236e-01$	$9.815509e-01$	$6.412054e-01$	$9.225543e-01$
$9.372086e-01$	$7.876298e-01$	$4.215441e-01$	$9.895819e-01$	$5.436153e-01$
$4.621722e-01$	$2.697455e-01$	$2.418158e-66$	$7.792889e-01$	$2.700199e-01$
$5.681294e-72$	$3.938088e-01$	$9.206758e-01$	$9.566921e-01$	$6.956183e-7$
$3.998902e-64$	$1.124886e-65$	$5.545532e-65$	$6.606923e-01$	$6.791583e-70$
$7.722793e-02$	$9.928220e-01$	$8.150995e-01$	$6.368642e-01$	$3.751953e-68$
$2.390559e-73$	$1.931186e-01$	$5.463323e-01$	$6.065751e-69$	$1.958048e-62$
$1.160810e-01$	$6.904894e-01$	$4.227690e-01$	$4.495270e-01$	$9.245160e-73$
$2.135549e-01$	$2.569015e-67$	$3.053534e-01$	$3.740873e-01$	$1.476473e-01$
$1.607731e-01$	$1.607616e-02$	$7.897918e-68$	$2.661928e-71$	$1.783588e-01$
$2.746740e-01$	$1.071445e-64$	$6.583901e-02$	$1.682636e-67$	$8.937961e-73$

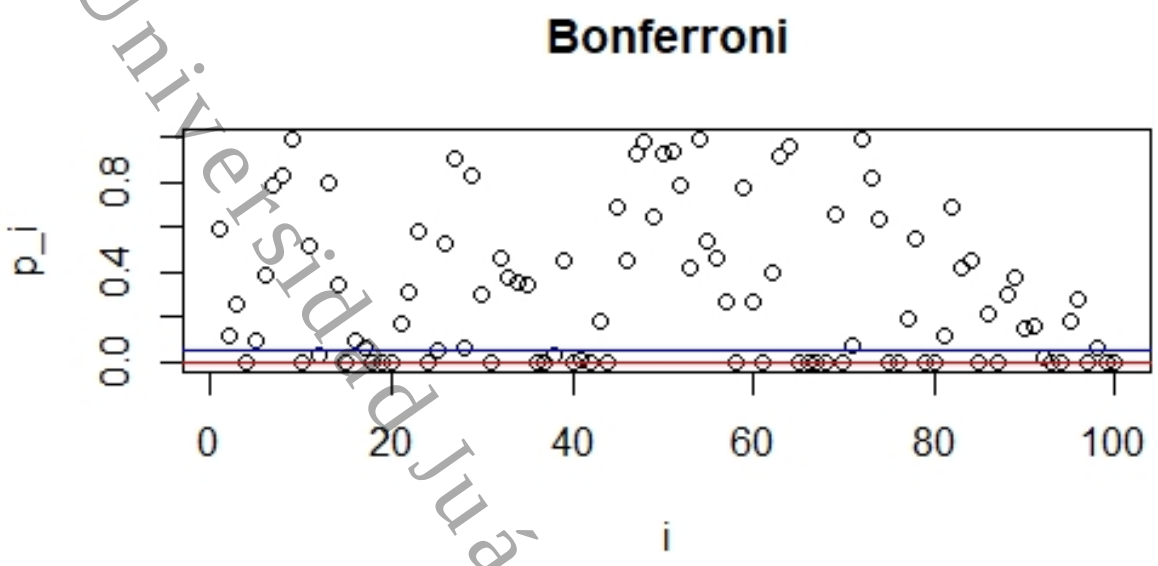

Figura 2.1: Dispersión de los $p-valor$ asociado a la prueba de hipótesis, donde de la línea azul marca la recta $\alpha = 0.05$ y la línea roja es el valor $5 * 10^{-4}$.

Entonces el método de Bonferroni rechaza la hipótesis nula para las p_i que están por debajo de la línea roja. Los $p-valores$ que están por debajo de esta línea roja son:

$4.091238e-69$	$2.650218e-68$	$2.579670e-67$	$1.183116e-65$	$3.924809e-72$
$5.091490e-66$	$3.244553e-70$	$8.790030e-69$	$3.773595e-68$	$1.881122e-69$
$3.681254e-64$	$1.226016e-68$	$2.418158e-66$	$5.681294e-72$	$6.956183e-71$
$3.998902e-64$	$1.124886e-65$	$5.545532e-65$	$6.791583e-70$	$3.751953e-68$
$2.390559e-73$	$6.065751e-69$	$1.958048e-62$	$9.245160e-73$	$2.569015e-67$
$7.897918e-68$	$2.661928e-71$	$1.071445e-64$	$1.682636e-67$	$8.937961e-73$

Podemos notar que estos $p-valores$ seleccionados por el método de Bonferroni, corresponden a las variables que sí son distintas de cero.

2.3.2. Procedimiento modificado de Bonferroni

Holm (1979) discutió otro "procedimiento modificado de Bonferroni".

Sean $p_{(1)}, p_{(2)}, \ldots, p_{(N)}$ los p-valores ordenados de las pruebas de hipótesis $H_0 = \{H_{(1)}, \ldots, H_{(N)}\}$. La H_i es rechazada si

$$p_{(j)} \leq \alpha/(N - j + 1)$$

para toda $j = 1, 2, \ldots, i$.

Sea I_0 el conjunto de índices correspondientes a las hipótesis nulas verdaderas (desconocidas) y supongamos que tiene N_0 elementos.

Definamos el evento:

$$A = \{p_{(i)} \leq \alpha/N_0 \text{ para algún } i \in I_0\}.$$

Así tenemos que

$$P(A) \leq \alpha \qquad \text{(por la desigualdad de Bonferroni).}$$

Por lo tanto, la probabilidad de rechazar alguna hipótesis verdadera es a lo más α. Es decir, Bonferroni modificado también controla la tasa de error familiar (FWER).

Ejemplo 2.3.2 *Aplicando el procedimiento modificado a los $p-$valores del ejemplo 2.3.1, tenemos lo siguiente:*

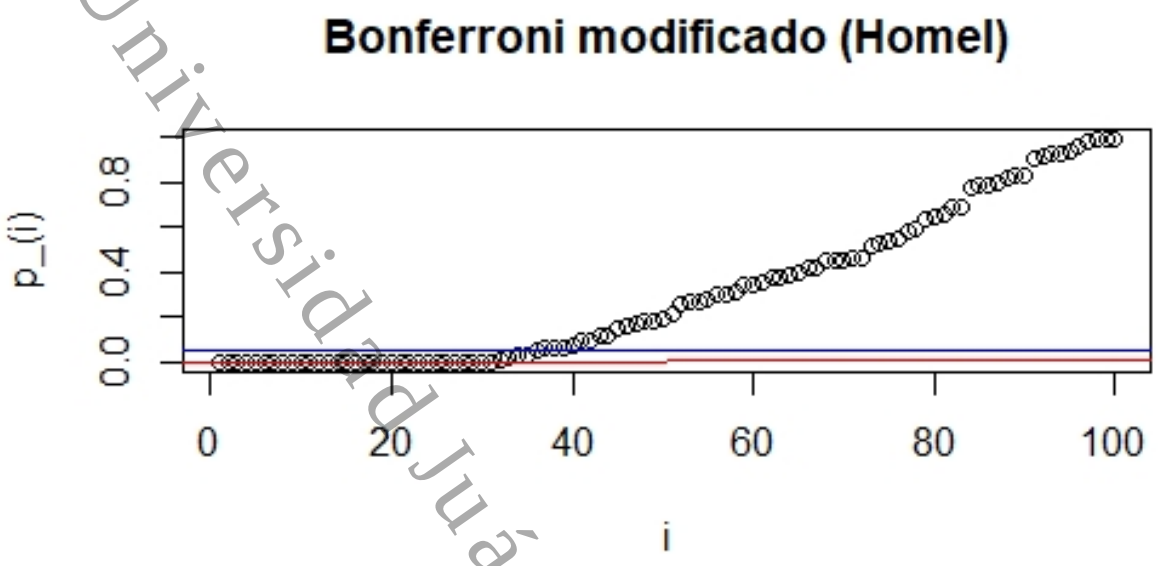

Figura 2.2: Dispersión de los $p-valor$ asociado a la prueba de hipótesis, donde la línea azul marca la recta $\alpha = 0.05$ y la curva roja es el valor $\alpha/(N-j+1)$.

Entonces el procedimiento de Bonferroni modificado rechaza la hipótesis nula para las $p_{(i)}$ que están por debajo de la recta roja. Los $p-valore$ que están por debajo de esta línea roja son:

$4.091238e-69$ $2.650218e-68$ $2.579670e-67$ $5.589601e-04$ $1.183116e-65$
$3.924809e-72$ $5.091490e-66$ $3.244553e-70$ $8.790030e-69$ $3.773595e-68$
$1.881122e-69$ $3.681254e-64$ $1.226016e-68$ $2.418158e-66$ $5.681294e-72$
$6.956183e-71$ $3.998902e-64$ $1.124886e-65$ $5.545532e-65$ $6.791583e-70$
$3.751953e-68$ $2.390559e-73$ $6.065751e-69$ $1.958048e-62$ $9.245160e-73$
$2.569015e-67$ $7.897918e-68$ $2.661928e-71$ $1.071445e-64$ $1.682636e-67$
$8.937961e-73$.

Podemos notar que los $p-valores$ que seleccionó el procedimiento de Bonferroni modificado corresponden a las variables distinta de cero, excepto por la que está de color rojo.

2.3.3. Algoritmo del control del FDR de Benjamini y Hochberg

Asumamos que nuestra regla de decisión D produce un p-valor p_i para cada caso i, tal que p_i tiene una distribución uniforme si H_{0i} es correcta,

$$H_{0i} : p_i \sim U(0, 1).$$

Denotamos a los valores ordenados de p_i por

$$p_{(1)} \leq p_{(2)} \leq \ldots \leq p_{(i)} \leq \ldots \leq p_{(N)}.$$

Siguiendo la notación de la tabla 2.1, sea R_D el número de casos rechazados, a_D el número de aquellos que en realidad son nulos y Fdp_D la proporción de descubrimientos falsos

$$Fdp_D = a_D/R_D \quad [\text{la cual es 0 si} \quad R_D = 0]$$

El algoritmo de Benjamini-Hochberg (BH) usa esta regla para un valor fijo q en $(0, 1)$. Sea i_{max} el mayor índice para el cual

$$p_{(i)} \leq \frac{i}{N} q,$$

y rechazar H_{0i}, la hipótesis nula correspondiente a $p_{(i)}$, si

$$i \leq i_{max},$$

no rechazar H_{0i} en caso contrario.

El siguiente teorema y su demostración se encuentran en [Efron, 2012], página 48.

Teorema 2.3.3 *Si los p-valores correspondientes a las hipótesis nulas correctas son independientes unos de otros, entonces la regla BH(q) basada en el algoritmo de BH controla la proporción esperada de descubrimientos falsos en q,*

$$E\left[Fdp_{BH(q)}\right] = \pi_0 q \leq q, \quad \textit{donde} \quad \pi_0 = N_0/N.$$

Demostración 2.3.4 *Sea $t \in (0, 1]$ y definamos*

$$R(t) = \#\{p_i \leq t\},$$

$$a(t) = \#\ de\ H_0\ 's\ realmente\ nulas\ tales\ que\ p_i \leq t,$$

$$FDR(t) = \frac{a(t)}{\text{máx}\{R(t), 1\}},$$

$$Q(t) = \frac{Nt}{\text{máx}\{R(t), 1\}}$$

y

$$t_q = \sup\{t : Q(t) \leq q\}.$$

Note que

$$R(p_{(i)}) = \#\{p_i \leq p_{(i)}\} = i$$

y

$$Q(p_{(i)}) = \frac{Np_{(i)}}{\text{máx}\{R(p_{(i)}), 1\}} = \frac{Np_{(i)}}{\text{máx}\{i, 1\}} = \frac{Np_{(i)}}{i}.$$

Esto implica que la regla de BH puede reescribirse como, rechazar $H_{0(i)}$ si $p_{(i)} \leq t_q$.
Supongamos que los p-valores asociados a las N_0 hipótesis realmente nulas, son independientes y uniformes. Por la Sección A.1, si

$$A(t) = \frac{a(t)}{t}, \qquad entonces \quad tA(t) \sim Bin(N_0, t).$$

Así tenemos que $E[A(t)] = \frac{1}{t}N_0 t = N_0$.
Podemos ver que para $s < t$ se tiene que

$$E[A(s) \mid A(t)] = A(t).$$

Entonces $A(t)$ es una martingala conforme t decrece de $t = 1$ a $t = 0$.

Del teorema de paro óptimo para martingalas (teorema A.2.1) tenemos que

$$E[A(t_q)] = E[A(1)],$$

entonces

$$E[A(t_q)] = E[A(1)] = E[1A(1)] = N_0.$$

Ahora bien, notemos que

$$\text{máx}\{R(t_q), 1\} = \frac{Nt_q}{Q(t_q)} = \frac{a(t_q)}{FDP(t_q)},$$

entonces

$$FDP(t_q) = \frac{a(t_q)}{Nt_q} Q(t_q).$$

Podemos ver que $Q(t_q) = q$ *(véase la Sección A.3 del Apéndice A), así tenemos que*

$$FDP(t_q) = \frac{a(t_q)}{Nt_q} q.$$

Finalmente tenemos que

$$E[FDP(t_q)] = E\left[\frac{a(t_q)}{Nt_q} q\right] = \frac{q}{N} E\left[\frac{a(t_q)}{t_q}\right] = \frac{q}{N} E[A(t_q)] = \frac{q}{N} N_0 \equiv \pi_0 q.$$

■

La proporción de casos nulos $\pi_0 = N_0/N$, aunque en la práctica es desconocida, veremos que puede ser estimada, y entonces q generalmente se cita como la tasa de control de BH(q).

Ejemplo 2.3.5 *Aplicando el algoritmo de Benjamini y Hochberg a los* $p-$*valores del ejemplo 2.3.1, tenemos lo siguiente:*

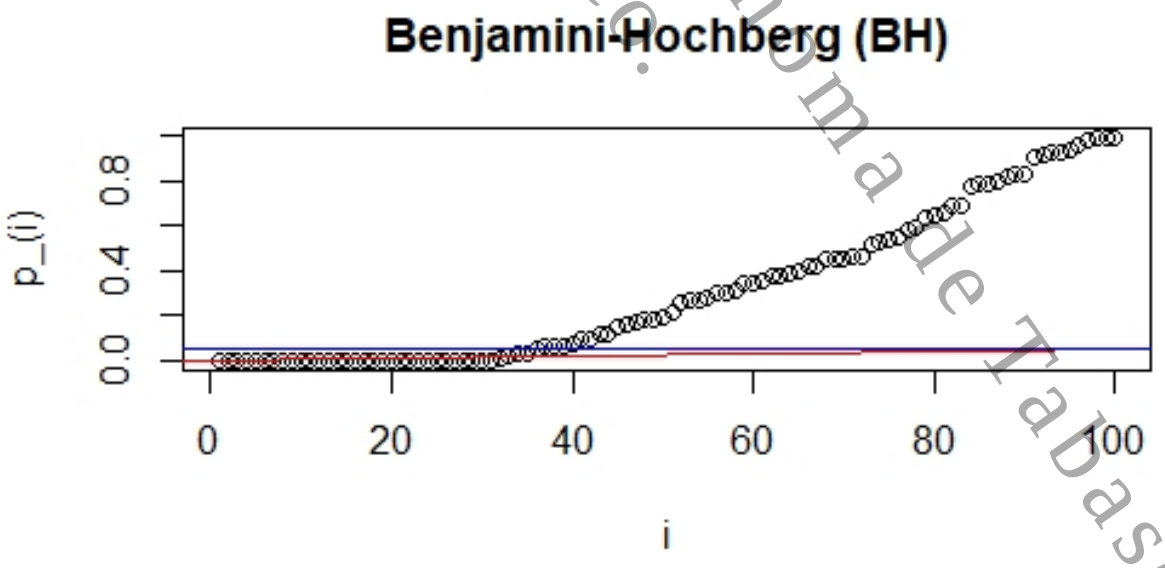

Figura 2.3: Dispersión de los $p-$*valor* asociado a la prueba de hipótesis, donde la línea azul marca la recta $\alpha = 0.05$ y la recta roja es el valor $j\alpha/N$.

Entonces el algoritmo de Benjamini y Hochberg no rechaza la hipótesis nula para las $p_{(i)}$ que están por debajo de la recta roja. Los $p-valore$ que están por debajo de esta línea roja son:

$4.091238e-69$ $2.650218e-68$ $2.579670e-67$ $5.589601e-04$ $1.183116e-65$
$3.924809e-72$ $5.091490e-66$ $3.244553e-70$ $8.790030e-69$ $3.773595e-68$
$1.881122e-69$ $9.464218e-03$ $3.681254e-64$ $1.226016e-68$ $2.418158e-66$
$5.681294e-72$ $6.956183e-71$ $3.998902e-64$ $1.124886e-65$ $5.545532e-65$
$6.791583e-70$ $3.751953e-68$ $2.390559e-73$ $6.065751e-69$ $1.958048e-62$
$9.245160e-73$ $2.569015e-67$ $1.607616e-02$ $7.897918e-68$ $2.661928e-71$
$1.071445e-64$ $1.682636e-67$ $8.937961e-73$.

Podemos notar que los p-valores seleccionados por el algoritmo de Benjamini-Hochberg corresponden a las variables distinta de cero, excepto por los tres p-valores que están de color rojo.

2.4. Error tipo S

Las definiciones y resultados que se presentan continuación fueron obtenidos de [Gelman and Tuerlinckx, 2000].

Los procedimientos de comparación clásicos están basados en calibrar el error de tipo I, es decir, la probabilidad de afirmar que $\theta_1 \neq \theta_2$ cuando de hecho $\theta_1 = \theta_2$. Por ejemplo, si realizamos el procedimiento de afirmar que $\theta_1 \neq \theta_2$ si el intervalo de confianza del 95 % para $\theta_1 - \theta_2$ excluye al cero, entonces nuestra tasa de error deberá ser a lo más 5 %, pero también podemos pensarlo en término del signo de la comparación, así, si obtenemos un intervalo de confianza del 95 % para $\theta_1 - \theta_2$ que es completamente positivo, "afirmamos con confianza" que $\theta_1 > \theta_2$, con un intervalo de confianza todo negativo "afirmamos con confianza" que $\theta_1 < \theta_2$, y con un intervalo de confianza que contiene al cero no "afirmamos con confianza" nada. Este procedimiento de comparación de signo es calibrado usando el error tipo S (por signo), el cual corresponde a identificar el error de signo de una comparación, es decir, afirmar que $\theta_1 > \theta_2$ cuando de hecho $\theta_1 < \theta_2$. Desde la perspectiva de la comparación de signo, creemos que la tasa de error tipo S es de interés más directo que la tasa de error tipo I.

2.4.1. Prueba de un lado, prueba de dos lados y la interpretación de los intervalos de confianza

La prueba clásica de uno y dos lados, prueba las hipótesis $\theta_j < \theta_k$ y $\theta_j \neq \theta_k$ respectivamente. Ya que se prueban las dos desigualdades $\theta_j < \theta_k$ y $\theta_k < \theta_j$, nuestro procedimiento de concentrarnos en el signo del intervalo de confianza corresponde a dos pruebas unilaterales simultáneas. Esta interpretación podría ser aceptable, pero preferimos pensar en las "afirmaciones con confianza" y el error tipo S surgiendo de la interpretación natural del intervalo del 95 % de confianza. Cuando el intervalo para $\theta_j - \theta_k$ incluye al cero, entonces es estándar decir que los dos parámetros no son estadísticamente diferentes. Cuando el intervalo para $\theta_j - \theta_k$ excluye al cero, entonces es estándar aceptar la diferencia como real y trabajar, con confianza, con la suposición de que el signo de la verdadera diferencia es como la dada por la diferencia estimada. Este procedimiento (de hacer afirmaciones confiables sobre el signo de una comparación si el intervalo del 95 % para la diferencia excluye al cero) es estándar en aplicaciones de regresión lineal, modelos lineales generalizados y análisis estadísticos más complicados, así como comparaciones de medias. Por lo tanto, no nos vemos a nosotros mismos evaluando una prueba de un lado o dos lados, sino más bien como evaluando el procedimiento estadístico estándar basado en la ubicación de un intervalo del 95 % con respecto al cero.

2.4.2. Prueba de hipótesis y error tipo S

Las definiciones y resultados que se presentan a continuación, fueron obtenidos de [Gelman and Carlin, 2014] y [Lu et al., 2019].

Consideremos la prueba de que μ no es cero, es decir,

$$H_0 : \mu = 0 \quad \text{vs.} \quad H_1 : \mu \neq 0.$$

Asumimos que el estadístico de prueba sigue una distribución normal $Z|\mu, \sigma \sim N(\mu, \sigma^2)$. La prueba de hipótesis tradicional se concentra sobre los siguientes dos errores estadísticos:

1. Tipo I: Rechazar la hipótesis nula H_0, cuando esta es verdadera.

2. Tipo II: No rechazar la hipótesis nula H_0, cuando esta no es verdadera.

La clave de las pruebas de hipótesis tradicionales es el nivel de significancia α, la cual controla la tasa de error de tipo I. Más específicamente, sea $z_\alpha = \Phi^{-1}(1-\alpha/2)$ el valor crítico de dos lados y rechazamos la hipótesis nula H_0 cuando $\mid Z \mid > z_\alpha$, así

$$P(\mid Z \mid > z_\alpha \mid H_0) = \alpha.$$

Aunque los errores tipo I/II son sin duda las piedras angulares del marco de pruebas de significación de la hipótesis nula, Gelman y Carlin (2014) argumentaron que el control de estos dos errores es insuficiente para capturar completamente el riesgo de los análisis de pruebas de significación de la hipótesis nula. Al darse cuenta de este peligro potencial, propusieron centrarse en dos nuevos errores:

1. Tipo S: El estadístico de prueba Z está en la dirección opuesta al tamaño del efecto μ, dado que es estadísticamente significativo.

2. Tipo M: El estadístico de prueba Z exagera el tamaño del efecto μ, dado que es estadísticamente significativo.

Bajo el marco de referencia clásico de las pruebas de hipótesis múltiples, para evaluar la credibilidad de una prueba de hipótesis calculamos la función potencia, la cual es una función de μ y se define formalmente como la probabilidad de rechazar la hipótesis nula H_0 cuando μ no es cero, es decir H_0 es falsa,

$$p = P(\mid Z \mid > \sigma z_\alpha \mid \mu). \tag{2.3}$$

Por definición, la función potencia es uno menos la probabilidad del error tipo II, por lo tanto con una potencia pequeña es más probable que el error tipo II ocurra.

Sin pérdida de generalidad asumimos que $\mu > 0$, entonces **la probabilidad de error tipo S** es:

$$s = P(Z < 0 \mid \mu, \mid Z \mid > \sigma z_\alpha). \tag{2.4}$$

Teorema 2.4.1 *Sean $\Phi(\cdot)$ y $\phi(\cdot)$ la función de distribución acumulada y la función de densidad de probabilidad de la distribución normal estándar, res-*

pectivamente, y $\lambda = \mu/\sigma$*. Las expresiones cerradas para la potencia y el error tipo S son:*

$$p = \Phi(-z_\alpha - \lambda) + 1 - \Phi(z_\alpha - \lambda) \tag{2.5}$$

y

$$s = \frac{\Phi(-z_\alpha - \lambda)}{\Phi(-z_\alpha - \lambda) + 1 - \Phi(z_\alpha - \lambda)}, \tag{2.6}$$

respectivamente.

2.4.3. Error tipo S y FDR direccional

Las definiciones y resultados que se presentan a continuación, fueron obtenidos de [Barber and Candès, 2019].

El enfoque del error tipo S es extremadamente relevante para nuestros problemas en regresión (posible dimensión alta) y en nuestra meta de encontrar los efectos importantes. Cuando elegimos reportar una variable, deberíamos ser capaces de afirmar con confianza su dirección de efecto, es decir, si $\beta_j > 0$ ó $\beta_j < 0$. Para insistir en este punto, seguramente tendríamos un poco de fe en un procedimiento que controlara el error tipo I, pero este no podría decirnos la dirección de los efectos. Otro aspecto útil del error tipo S es que efectos pequeños, $\beta_j \approx 0$, son generalmente aquellos para los que no podemos tener mucha certeza sobre su dirección, ya que el tamaño del efecto es comparable o inferior al nivel de ruido de nuestra estimación. Por lo tanto, si estuviéramos preocupados por el error de signo probablemente no reportaremos este efecto como descubrimiento (incluso si creemos que ningún efecto es exactamente cero).

Para medir nuestro éxito al seleccionar solo estos efectos que son lo suficientemente grandes para ser significativamente distintas del ruido, nos gustaría controlar la tasa de descubrimientos falsos de dirección mixta (FDR_{dir}), definida como sigue, sea $\widehat{S} = \{1, \ldots, p\}$ el conjunto de variables seleccionadas junto con la estimación $\hat{sign}_j \in \{\pm 1\}$ de la estimación de los efectos,

$$FDR_{dir} = E[FDP_{dir}],$$

donde la proporción de descubrimientos falsos de direcciones mixtas (FDP_{dir}) está dada por

$$FDP_{dir} = \frac{\mid \{j \in \widehat{S} : \widehat{sign}_j \neq sign(\beta_j)\} \mid}{\mid \widehat{S} \mid \vee 1}, \tag{2.7}$$

con la convención de que $sign(0) = 0$. Esta definición incluye el error de tipo I y tipo S; se produce un error o un descubrimiento falso cuando se selecciona un efecto cero, o cuando un efecto no cero es seleccionado pero con el signo incorrecto. El término "mixtas" viene de esta combinación de los dos tipos de errores, para nuestro propósito, consideramos declarar que $\beta_j > 0$ es un error de signo si la verdad es que $\beta_j = 0$ o $\beta_j < 0$, por lo que no distinguirá entre estos dos tipos de errores y así abandonaremos el término "mixtas" y nos referiremos a esta simplemente como "FDR direccional". El FDP direccional es entonces el número total de errores de estos tipos (combinados) entre el conjunto seleccionado. En conjuntos de regresión en el cual algunas de las β_j's son aproximadamente cero, pero quizás no exactamente cero, controlar el FDR direccional puede ser más apropiado que el FDR clásico.

Notemos que el control del FDR direccioal tiene un sentido diferente al del FDR clásico, ya que la significancia no es calibrada usando el error tipo I, el cual implica asumir que algunas de las β_j's son ceros. En contraste, buscamos un origen de inferencia que se sostiene sin importar el valor de las β_j's. Debe quedar claro que la noción equivalente (2.7) asociada al error tipo S es en principio más difícil de controlar, ya que por definición, tenemos que

$$FDR_{dir} \geq FDR,$$

debido a que el FDR clásico no registra un error cuando la variable j-ésima es incluida correctamente en el modelo pero con signo incorrecto, mientras que la FDR direccional lo hace.

Ejemplo 2.4.2 *Para ilustrar la aplicación del FDR_{dir} utilizaremos la regresión lineal del ejemplo 1.5.1. Los valores verdaderos de las 30 β's distintas de cero son* 5 *y las demás son ceros.*

Los valores estimados de los β's seleccionados por el método de Stepwise son:

5.853672 8.413910 3.463552 2.425703 5.189199
6.053700 1.736408 5.642772 4.786781 4.312202
4.732620 4.761029 5.044814 7.035653 −1.208749
1.577919 5.290481 3.882444 4.516975 3.865281
4.287953 3.601286,

los de color rojo corresponden a las β's que realmente son ceros.

El signo de los valores de color rojo son $+1, -1$ y $+1$ respectivamente, mientras que su signo verdadero es 0, por otra parte el signo de los demás valores es $+1$, que corresponde a su signo verdadero, así

$$FDR_{dir_{Stepwise}} = \frac{3}{22} \geq FDR_{Stepwise}.$$

Por otro lado, los valores de β estimados por la regresión Lasso son:

4.944207431 0.021757416 4.771129097 5.072456980 0.083269775
5.161641946 4.780627157 5.019850248 0.001435765 0.214540592
0.018742928 4.963327418 0.000142423 4.840454832 4.971011350
0.055922572 0.011736179 4.830721292 4.749276238 4.843873687
4.874263362 4.931928218 4.898254990 5.122573574 4.934583000
4.813188785 4.984562391 4.888936605 5.070724195 0.006356862
5.048082439 4.783468978 5.089163295 0.118396407 4.800765696
0.011891637 4.850059547 4.920012235 0.149672912 4.831705392
0.081747283 4.872851045 4.930373262.

El signo de los valores de color rojo es $+1$, mientras que su signo verdadero es 0, por otra parte el signo de los demás valores también es $+1$, que corresponde a su signo verdadero, así

$$FDR_{dir_{Lasso}} = \frac{13}{43} \geq FDR_{Lasso}.$$

Capítulo 3

Metodología de Knockoff

3.1. Introducción

En este capítulo, exploraremos un enfoque innovador y prometedor en el campo de la selección de características y la inferencia estadística, la teoría de "knockoffs". A lo largo de este capítulo, veremos los detalles de esta teoría, la cual tiene como objetivo la selección de variables controlando la tasa de descubrimientos falsos (FDR), su enfoque se basa en la construcción de knockoffs (réplicas falsas) de las variables originales, que se utilizan para estimar el nivel de importancia de cada variable en relación con el objetivo, teniendo en cuenta tanto la relación lineal como las correlaciones existentes entre las variables.

Las definiciones y resultados que se presentan a continuación fueron obtenidos en [Barber and Candès, 2015] y [Barber and Candès, 2019].

3.2. Knockoffs

Supongamos que tenemos una variable de respuesta y y algunas variables explicativas potenciales X_j sobre n observaciones. Nuestras observaciones obedecen al clásico modelo de regresión lineal

$$y = X\beta + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I), \qquad n \geq p, \tag{3.1}$$

donde $y \in \mathbb{R}$ es un vector de respuesta, $X \in \mathbb{R}^{n \times p}$ es una matriz de diseño conocida, $\beta \in \mathbb{R}^p$ es un vector de coeficientes desconocidos y $\epsilon \sim N(0, \sigma^2 I)$ es un ruido gaussiano. Como estamos interesados en una inferencia válida con un número finito de muestras, por ahora nos restringiremos al caso donde

$n > p$, de otra manera el modelo podría no ser siempre identificable.

Informalmente el FDR es la proporción esperada de variables falsamente seleccionadas, un descubrimiento falso es una variable que no está en el modelo verdadero. Formalmente, el FDR de un procedimiento de selección que regresa un subconjunto $\widehat{S} \subset \{1, \ldots, p\}$ de variables es definido como

$$\text{FDR} = E\left[\frac{\#\{j : \beta_j = 0,\ j \in \widehat{S}\}}{\#\{j : j \in \widehat{S}\} \vee 1}\right] \tag{3.2}$$

donde $a \vee b = \max\{a, b\}$ y definimos $\widehat{S} = \emptyset$ en caso de seleccionar cero variables. Diremos que una regla de decisión controla el FDR a un nivel q si este FDR es a lo más q, sin importar el valor de los coeficientes de β.

Por otro lado, la **potencia** de un procedimiento de selección de variables se define como

$$P = E\left[\frac{\#\left\{j : \beta_j \neq 0 \text{ y } j \in \widehat{S}\right\}}{k}\right], \tag{3.3}$$

donde k es el total de β_j's distintas de cero. La potencia representa la proporción esperada de descubrimientos verdaderos.

El filtro de knockoffs

Se introducirá un procedimiento de control de FDR general que esté garantizado para trabajar bajo la matriz de diseño $X \in \mathbb{R}^{n \times p}$, siempre que $n \geq p$ y la respuesta y sigue un modelo lineal gaussiano como en (3.1). Una característica importante de este procedimiento es que no requiere ningún conocimiento del nivel de ruido σ, además no asume ningún conocimiento sobre el número de variables en el modelo, el cual puede ser arbitrario. Ahora describiremos los pasos de este método.

Paso 1: Construcción del knochoffs

Para cada característica X_j en el modelo (es decir, las columnas de X), construiremos una característica "knockoff" $\widehat{X}_j$. El objetivo de las variables knockoff es imitar la estructura de correlación de las variables originales de una

manera muy específica que permita el control del FDR.

Específicamente, para construir log knockoffs, primero calcularemos la matriz de Gram $\Sigma = X^T X$ de las variables originales, después normalizamos cada variable para que $\Sigma_{jj} = \|X_j\|_2^2 = 1$ para toda j. Aseguramos que estas variables knockoff cumplen

$$\widehat{X}_j^T \widehat{X} = \Sigma, \quad X^T \widehat{X} = \Sigma - diag(s), \tag{3.4}$$

donde s es un vector p-dimensional no negativo. En otras palabras, $\widehat{X}$ exhibe la misma estructura de covarianza como el diseño original X, pero además, las correlaciones entre variables originales y knockoff son la misma como aquellas entre las originales (porque σ y $\sigma - diag(s)$ son iguales en las entradas fuera de la diagonal),

$$X_j^T \widehat{X}_k = X_j^T X_k, \quad \forall j \neq k.$$

Sin embargo, comparando una variable X_j con su variable knockoff $\widehat{X}_j$, vemos que

$$X_j^T \widehat{X}_j = \Sigma_{jj} - s_j = 1 - s_j,$$

mientras que $X_j^T X_j = \hat{X}_j^T \hat{X}_j = 1$. Para asegurar que el método tiene una buena potencia estadística, veremos que debemos escoger las entradas de s tan grande como sea posible para que la variable X_j no sea tan similar a su knockoff $\widehat{X}_j$.

Una estrategia para la construcción de $\widehat{X}$ es escoger $s \in \mathbb{R}_+^p$ que satisfaga $diag(s) \preceq 2\Sigma$ y construir la matriz $\widehat{X}$ de $n \times p$ de variables knockoff como:

$$\widehat{X} = X(I - \Sigma^{-1} diag(s)) + \widehat{U} C, \tag{3.5}$$

donde $\widehat{U}$ es una matriz de $n \times p$ que es ortogonal al espacio generado por las variables X y $C^T C = 2diag(s) - diag(s)\Sigma^{-1} diag(s)$ es una descomposición de Cholesky.

Paso 2: Cálculo del estadístico para cada par de variables originales y knockoff

Ahora deseamos introducir estadísticos W_j para cada β_j, $j \in \{1, \ldots, p\}$, que nos ayudará a apartar las variables que están en el modelo de las que no lo

están. Estas W_j's son construidas de tal manera que valores positivos grandes son evidencia en contra de la hipótesis nula $\beta_j = 0$.
Ahora, consideraremos el modelo de regresión Lasso, una regresión con la penalización de la norma ℓ_1 que provee estimaciones dispersas de los coeficientes β, dadas por

$$\widehat{\beta}(\lambda) = argmin_b \left\{ \frac{1}{2}\|y - Xb\|_2^2 + \lambda\|b\|_1 \right\}. \tag{3.6}$$

Para modelos lineales dispersos, Lasso es conocido por ser asintóticamente preciso para la selección de variables y para la estimación de coeficientes significativos, e incluso en un escenario no asintótico, típicamente veremos que $\widehat{\beta}(\lambda)$ incluirá algunas variables significativas y pocas variables nulas con algún valor del parámetro de penalización λ. Tomamos Z_j como el valor λ en la regresión Lasso para el cual la variable X_j entra por primera vez al modelo,

$$Z_j = sup\{\lambda : \widehat{\beta}_j(\lambda) \neq 0\}, \tag{3.7}$$

el cual es grande para la mayoría de las variables significativas y pequeño para la mayoría de las variables nulas. Sin embargo, para ser capaces de cuantificar esto y elegir un umbral apropiado para seleccionar variables, necesitamos usar las variables knockoff para calibrar nuestro umbral. Con esto en mente, calculamos el estadístico (3.7)con la matriz de diseño aumentada $\left[X\widehat{X}\right]$ de $n \times 2p$, así que $\left[X\widehat{X}\right]$ reemplaza a X en (3.6). Esto produce un vector $2p$-dimensional $(Z_1, \ldots, Z_p, \widehat{Z}_1, \ldots, \widehat{Z})$. Finalmente, para cada $j \in \{1, \ldots, p\}$, definimos

$$W_j = \left(Z_j \vee \hat{Z}_j\right) \cdot \begin{cases} +1 & Z_j > \hat{Z}_j \\ -1 & Z_j < \hat{Z}_j \end{cases}, \tag{3.8}$$

(podemos tomar $W_j = 0$ en caso de que $Z_j = \widehat{Z}_j$). Un valor positivo grande de W_j indica que la variable X_j entró al modelo Lasso temprano y que lo hace antes que su copia $\widehat{X}$. Por lo tanto, esta es una indicación de que esta variable es significativa de forma genuina y pertenece al modelo. También podemos considerar otra alternativa para la construcción de los W_j's , por ejemplo, en lugar de registrar la entrada de las variables en el modelo de Lasso, podemos considerar métodos de selección hacia adelante y registrar el orden en el cual las variables son añadidas al modelo.

Paso 3: Calcular un umbral dependiente de los datos para los estadísticos

Deseamos seleccionar variables tales que W_j sea grande y positiva, es decir, tales que $W_j \geq t$ para alguna $t > 0$. Sea q el FDR objetivo y definamos un umbral T que depende de los datos de la siguiente manera:

$$T = \text{mín}\left\{t \in \mathfrak{W} : \frac{\#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} \leq q\right\}, \tag{3.9}$$

(o $T = +\infty$ si este conjunto es vacío).

donde $\mathfrak{W} = \{| W_j |: j = 1, \ldots, p\} \setminus \{0\}$ es el conjunto de valores distintos de cero alcanzados por los $| W_j |'$s (si $W_j = 0$ para alguna variable X_j, entonces éste no da evidencia para rechazar la hipótesis de que $\beta_j = 0$, entonces nuestro método podría nunca seleccionar esta variable). Veremos que la fracción anterior es una estimación de la proporción de falsos positivos si seleccionamos todas las variables j's con $W_j \geq t$. Por esta razón frecuentemente nos referiremos a esta fracción como la estimación knockoff de FDP.

A continuación definiremos el proceso de selección de variables.

Definición 3.2.1 (knockoff) *Construimos $\widehat{X}$ como en (3.5) y calculamos el estadístico W_j que satisface las propiedades de suficiencia y antisimetría. Entonces se selecciona el modelo*

$$\widehat{S} = \{j : W_j \geq T\}, \tag{3.10}$$

donde T es el umbral dependiente de los datos (3.9).

Teorema 3.2.2 *Para cualquier $q \in [0, 1]$, el método de knockoff satisface*

$$E\left[\frac{\#\left\{j : \beta_j = 0,\ j \in \widehat{S}\right\}}{\#\left\{j : j \in \widehat{S}\right\} + q^{-1}}\right] \leq q,$$

donde el valor esperado es tomado sobre el ruido gaussiano ϵ en el modelo (3.1), cuando se tratan a X y $\widehat{X}$ como fijos.

El "FDR modificado" delimitado por este teorema es muy cercano al FDR en escenarios donde un número grande de variable son seleccionadas, pero algunas veces puede ser preferible controlar el FDR exactamente. Por eso, a continuación se presenta un procedimiento un poco más conservador.

Definición 3.2.3 (knockoff +) *Seleccionamos un modelo como en la definición 3.2.1 pero con un umbral T que depende de los datos, definido como*

$$T = \text{mín}\left\{t \in \mathfrak{W} : \frac{1 + \#\{j : W_j \leq -t\}}{\#\{j : W_j \geq t\} \vee 1} \leq q\right\}, \tag{3.11}$$

$T = +\infty$ si este conjunto es vacío.

Notemos que el umbral T escogido en knockoff+ es siempre más grande (o igual) que el escogido en (3.9) por el filtro de knockoff, esto significa que knockoff+ es (ligeramente) más conservador.

Teorema 3.2.4 *Para cualquier $q \in [0, 1]$, el método knockoff+ satisface*

$$FDR = E\left[\frac{\#\left\{j : \beta_j = 0,\ j \in \widehat{S}\right\}}{\#\left\{j : j \in \widehat{S}\right\} \vee 1}\right] \leq q,$$

donde el valor esperado es tomado sobre el ruido Gaussiano ϵ en el modelo (3.1), cuando se tratan a X y $\widehat{X}$ como fijos.

Extensión a $p \leq n \leq 2p$

Cuando $n \leq 2p$, no podemos encontrar un subespacio de dimensión p el cual sea ortogonal a X, por lo que no podemos construir $\widehat{U}$ como antes. Aún podemos usar el filtro knockoff, sin embargo, mientras el nivel de ruido σ sea conocido o pueda ser estimado. Por ejemplo, bajo el modelo de ruido gaussiano (3.1), podemos usar el hecho de que la suma de los cuadrados de los residuales para el modelo completo es distribuida como $\|y - X\widehat{\beta}^{LS}\|_2^2 \sim \sigma\chi^2_{n-p}$, donde $\widehat{\beta}^{LS}$ es el vector de coeficientes de regresión por mínimos cuadrados. Ahora tomando $\widehat{\sigma}$ como la estimación de σ, obtengamos un vector $(2p - n)$dimensional y' con entradas i.i.d. $N(0, \widehat{\sigma}^2)$. Si $n - p$ es grande, entonces $\widehat{\sigma}$ será un estimación extremadamente precisa de σ y podremos proceder como si $\widehat{\sigma}$ y σ fueran exactamente iguales. Ahora aumentamos el vector de respuesta y con el nuevo vector y' de tamaño $(2p-n)$ y aumentamos la matriz de diseño X con $2p - n$ filas de ceros. Entonces aproximadamente,

$$\begin{bmatrix} y \\ y' \end{bmatrix} \sim N\left(\begin{bmatrix} X \\ 0 \end{bmatrix} \beta, \sigma^2 I\right).$$

Ahora tenemos un modelo lineal con p variables y $2p$ observaciones y entonces podemos aplicar el filtro knockoff a estos datos de filas aumentadas usando el método descrito para el caso $n \geq 2p$.

Ejemplo 3.2.5 *Para ilustrar la metodología de knockoff utilizaremos la regresión lineal dada en el ejemplo 1.5.1. Las variables seleccionadas por esta metodología son:*

$$\begin{array}{cccccccccc} 4 & 10 & 15 & 19 & 20 & 24 & 31 & 36 & 37 & 40 \\ 42 & 44 & 58 & 61 & 65 & 66 & 67 & 70 & 75 & 76 \\ 79 & 85 & 87 & 93 & 94 & 97 & 99 & 100. & & \end{array}$$

Podemos observar que todas estas variables seleccionadas por el método knockoff corresponden a variables con β's distintas de cero, pero no son todas.

También tenemos que

$$FDR_{knockoff} = \frac{0}{28} = 0.$$

Notemos que en este ejemplo se tiene que

$$FDR_{knocff} \leq FDR_{Stepwise} \leq FDR_{Lasso}.$$

3.3. Control del error tipo S

Aunque este es más difícil de controlar, ya hemos discutido que el error tipo S es más significativo para problemas en regresión, ya que nos gustaría ser capaces de poder decir la dirección del efecto con confianza (y debe evitar informar aquellas variables cuya dirección no podemos determinar de manera confiable).
Consideremos el escenario donde $n > 2p$ y $y \sim N(X\beta, \sigma^2 I)$, entonces para estimar la dirección del efecto (el signo de β_j), notemos que

$$(X_j - \widehat{X}_j)^T \sim N(s_j\beta_j, 2s_j\sigma^2) \quad \text{para} \quad j = 1, \ldots, p,$$

donde $s \geq 0$ es el de la construcción del knockoff en la ecuación (3.4). Por lo tanto, un estimador natural para el signo de β_j es

$$\widehat{sign}_j = sign((X_j - \widehat{X}_j)^T y). \tag{3.12}$$

Con esto, el filtro de knockoff sin ajuste alguno controla el FDR direccional así como el FDR. Para entender intuitivamente por qué esta medida de error, que en principio es más difícil de controlar que el FDR, todavía está acotada por el mismo método, notemos que para un efecto no negativo ($\beta_j \geq 0$), la probabilidad de elegir un signo negativo, es decir, $\widehat{sign}_j = -1$, es de hecho más alta cuando $\beta_j = 0$, este error es menos posible si $\beta_j > 0$ ya que en este caso tendríamos qu $(X_j - \widehat{X}_j)^T y$ se distribuye normalmente con una media positiva.

Teorema 3.3.1 *Asumamos que $y \sim N(X\beta, \sigma^2 I)$ y fijemos un nivel q para el FDR deseado. Con la dirección de los efectos estimados (3.12), knockoff+ controla el FDR direccional (2.7), es decir, $FDR_{dir} \leq q$, para el conjunto $\widehat{S}$ seleccionado. Si en su lugar usamos knockoff, entonces $mFDR_{dir} \leq q$, donde $mFDR_{dir}$ es la versión ligeramente modificada, definida como*

$$mFDR_{dir} = E\left[\frac{|\{j \in \widehat{S} : \widehat{sign}_j \neq sign(\beta_j)\}|}{|\widehat{S}| + q^{-1}}\right], \tag{3.13}$$

donde el conjunto seleccionado $\widehat{S}$ está dado por (3.10).

Este resultado es más fuerte que el teorema 3.2.2 y 3.2.4, que establecen que el knockoff+ controla el FDR a un nivel q, esto es $FDR \leq q$ y el knockoff controla el FDR modificado, es decir, el valor esperado del cociente entre el número de descubrimientos falsos $|\{j \in \widehat{S} \text{ y } \beta_j = 0\}|$ y $|\widehat{S}| + q^{-1}$.

3.4. Knockoff en dimensión alta

En dimensión alta, cuando $p > n$, la construcción de knockoff ya no es posible, de hecho, la condición de la matriz de Gram en (3.4) podría ser posible solo si $s = 0$, es decir, si $\widehat{X}_j = X_j$ para cada variable $j = 1, \ldots, p$, así que el procedimiento de knockoff tendría potencia cero. En este escenario, una aproximación sencilla, podría ser usar parte de los datos para reducir el número de variables y una parte disjunta de los datos para correr el filtro de knockoff.

Detección de variables y re-utilización de datos

Consideremos dividir las n observaciones en dos grupos disjuntos de tamaño n_0 y $n_1 = n - n_0$, usadas para la primera detección de un conjunto más pequeño de variables potencialmente relevantes, y posteriormente correr el procedimiento de selección del modelo sobre este conjunto reducido de variables, respectivamente. Denotaremos a estas dos particiones disjuntas de los datos como $(X^{(0)}, y^{(0)}) \in \mathbb{R}^{n_0 \times p} \times \mathbb{R}^{n_0}$ y $(X^{(1)}, y^{(1)}) \in \mathbb{R}^{n_1 \times p} \times \mathbb{R}^{n_1}$, posteriormente se siguen los siguientes dos pasos.

- *Paso de detección*: Usando $(X^{(0)}, y^{(0)})$, identificamos un subconjunto $\widehat{S}_0 \subset \{1, 2, \ldots, p\}$, de variables potencialmente relevantes, tal que $\mid \widehat{S}_0 \mid < n_1$.
- *Paso de selección*: Ignoramos todas las variables descartadas en el paso anterior, corremos el procedimiento knockoff con los datos restantes, es decir, con $(X^{(1)}_{\widehat{S}_0}, y^{(1)})$.

Este sencillo procedimiento de división de datos es una extensión natural del filtro knockoff en dimensión baja y es claro que el procedimiento controla el FDR direccional en el último paso de selección del modelo, siempre que el paso de selección capture correctamente todas las variables relevantes (aquellas con coeficiente de regresión distintas de cero), una propiedad a la que a menudo se hace referencia como filtrado seguro en la literatura, los falsos positivos en el paso de selección, incluyendo características nulas, no suponen ningún problema.

Existe una pérdida de potencia inherente debido a la división de datos, ya que el paso de selección del modelo utiliza n_1 en lugar de n observaciones. Sin embargo, el uso de conjuntos de datos disjuntos para los pasos de filtrado y selección es fundamental, ya que la distribución de $y^{(0)}$ no puede tratarse como un modelo lineal gaussiano una vez que ha tenido lugar el paso de filtrado. El paso de filtrado es una función de la variable aleatoria $y^{(0)}$, por lo que la siguiente modificación del paso de selección no controlaría el FDR, ignorando cualquier variable que fuera descartada en el paso de filtración, corremos el procedimiento knockoff sobre el conjunto completo de datos, es decir, $(X_{\widehat{S}_0}, y)$. La pérdida del control del FDR no es meramente teórica: una característica nula X_j que se elige en el paso de detección es generalmente

más probable que aparezca como un falso positivo cuando se ejecuta el filtro knockoff, lo que lleva a un FDR más alto.

Debido a esto, en [Barber and Candès, 2019] se proponen dos mecanismos para aumentar la potencia relativa al procedimiento de división de datos descrito anteriormente:

1. *Reciclaje de datos*, donde la primera parte de los datos divididos se pueden reutilizar hasta cierto punto sin perder nada del control del FDR garantizado.

2. *Estadística signada*, donde podemos decidir de manera adaptativa enfocar nuestra búsqueda solo en un efecto positivo o solo en un efecto negativo.

Ejemplo 3.4.1 *Para ilustrar la metodología de knockoff en dimensión alta utilizaremos la regresión lineal dada en el ejemplo 1.5.2. Las variables seleccionadas por esta metodología son:*

5	22	26	58	67	103	171	172	187	189
192	208	212	221	222	224	229	232	238	242
243	255	266	275	276.					

Podemos observar que todas estas variables seleccionadas por el método knockoff corresponden a variables con β's distintas de cero, excepto por las que están de color rojo.

Por lo que tenemos que

$$FDR_{knockoff} = \frac{5}{25} = 0.2.$$

Notemos que en este ejemplo se tiene que

$$FDR_{knockoff} \leq FDR_{Lasso} \leq FDR_{Stepwise}.$$

Capítulo 4

Simulaciones y ejemplo de aplicación

4.1. Introducción

Anteriormente se aplicaron los métodos de selección de variables en un mismo ejemplo simulado y se observó su comportamiento. En este capítulo se mostrarán los resultados de una serie de simulaciones de modelos lineales a los cuales se les aplicaron los métodos de selección de variables knockoff, Stepwise y Lasso. Posteriormente se mostrará un ejemplo de la metodología de knockoff a datos reales.

4.2. Simulaciones

Se realizó una serie de simulaciones en RStudio. Éstas consisten en generar 400 modelos lineales y aplicarles los métodos de selección de variables knockoff, Stepwise y Lasso, para algunos valores del número de observaciones (n) y el número de variables (p), y se calculó la media del FDR y la media de la potencia (P) de cada método, siendo estas medias las versiones muestrales de (3.2) y (3.3), respectivamente.

Las variables utilizadas en las simulaciones se tomaron de la siguiente manera:

- La matriz X es una matriz de $n \times p$, donde las filas son vectores independientes idénticamente distribuidos $N_p(\mathbf{0}, \Sigma)$, donde

$$\Sigma = \text{toeplitz}(1, \rho, \rho^2, \ldots, \rho^{p-1}),$$

  con $0 < \rho < 1$.

- Para el vector β, tomamos un vector p-dimensional v_k que contiene k entradas iguales a 1, elegidas aleatoriamente, y el resto de las entradas iguales a 0. Tomamos $A \in \mathbb{R}$, con $A \neq 0$, a la cual llamamos *amplitud* y definimos β como:

$$\beta = \frac{A}{\sqrt{n}} v_k.$$

- El vector Y se define de la siguiente manera:

$$Y = X\beta + \epsilon,$$

donde ϵ es un vector aleatorio con distribución normal estándar n-variada.

Las simulaciones se realizaron para 10 valores diferentes de alguno de los tres parámetros: amplitud (A), el número de betas distintas de cero (k) y el parámetro ρ. Se mantuvo fijo el valor de los otros dos parámetros. Se consideraron los tres casos: $p \leq n$, $p \leq n \leq 2p$ y $n \ll p$.

4.2.1. Caso I: $p \leq n$

Para el caso $p < n$ fijamos $p = 1500$ y $n = 3000$. Veremos el comportamiento del FDR y la potencia al modificar la amplitud, el número de betas distintas de cero o el parámetro ρ.

Efecto de la amplitud

La amplitud tomará los siguientes valores $A = 20, 40, 60, \ldots, 200$, fijamos $k = 30$ y $\rho = 0.3$.

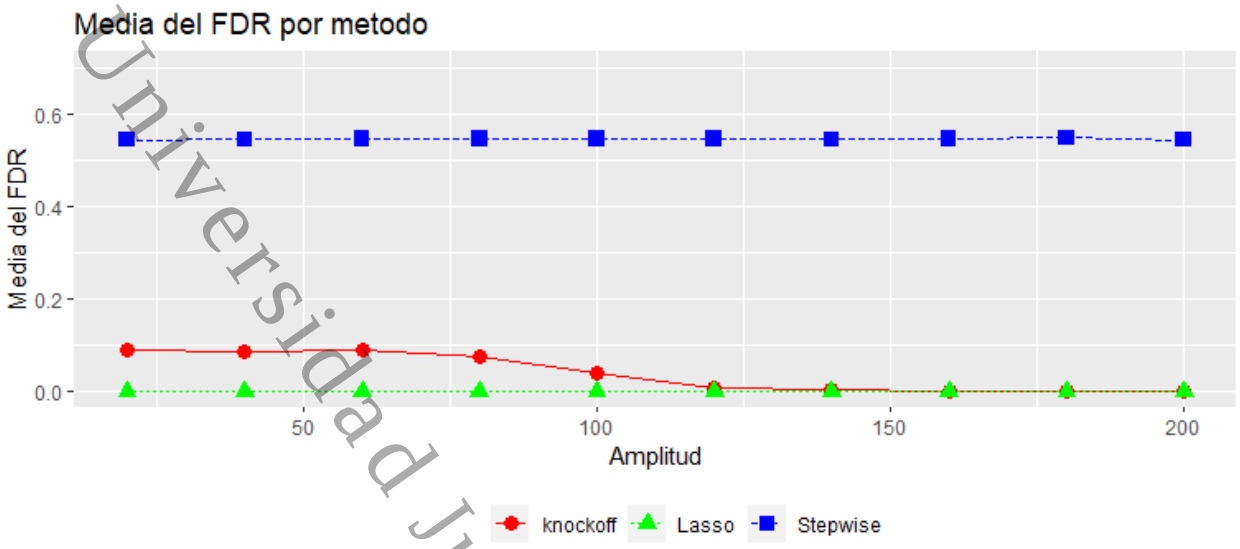

Figura 4.1: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 3000$, $p = 1500$ y $k = 30$, $\rho = 0.3$

Podemos observar que si la amplitud crece, el FDR para el método knockoff disminuye y para los métodos de Lasso y Stepwise la variación de la amplitud, en este caso, no parece tener ningún efecto. Lasso y knockoff son los métodos con FDR más bajo.

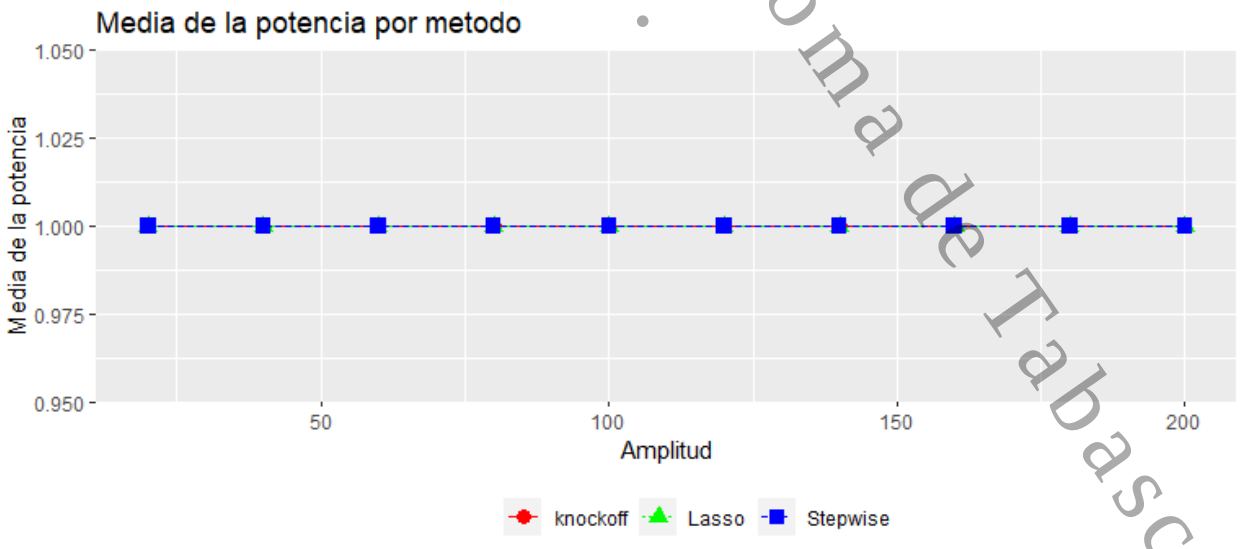

Figura 4.2: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 3000$, $p = 1500$ y $k = 30$, $\rho = 0.3$

Por otro lado, notamos que la potencia de los tres métodos, en este caso, es 1, es decir, que los tres métodos seleccionaron todas las variables verdaderas.

Efecto del aumento de la variable k

Ahora el número de betas distintas de cero tomará los valores $k = 10, 20, 30, \ldots, 100$, fijando $A = 50$ y $\rho = 0.3$.

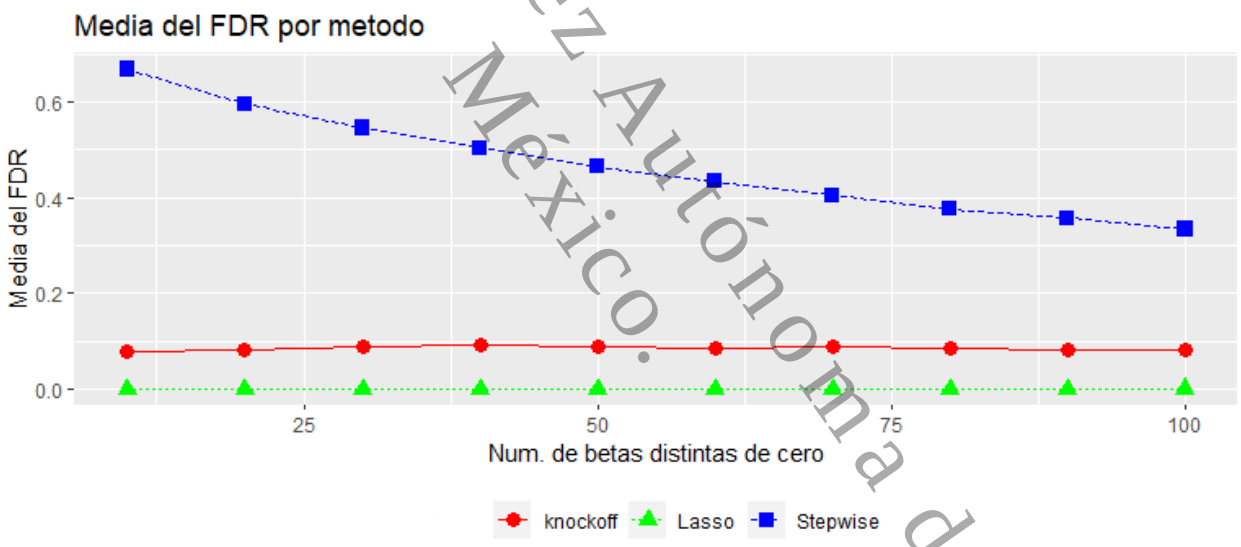

Figura 4.3: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando el número de betas distintas de cero k, con $n = 3000$, $p = 1500$ y $A = 50$, $\rho = 0.3$

Observamos que la variación de k no tiene un efecto significativo para el knockoff y Lasso, en cambio para Stepwise mientras k crece su FDR decrece. Nuevamente, Lasso y knockoff tienen FDR más bajo.

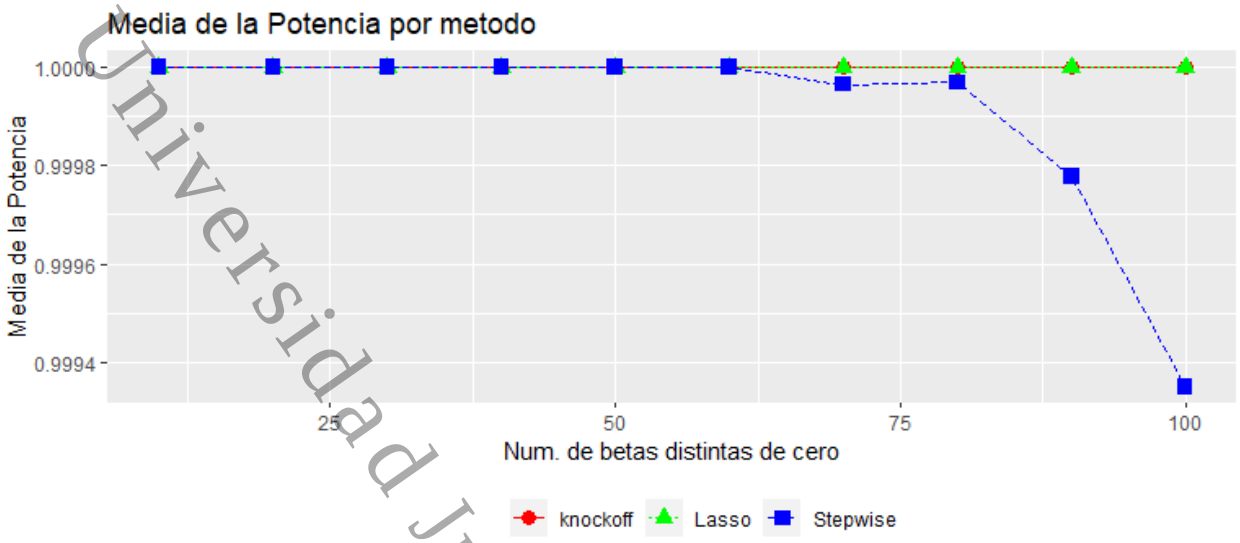

Figura 4.4: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando el número de betas distintas de cero k, con $n = 3000$, $p = 1500$ y $A = 50$, $\rho = 0.3$

Por otro lado la potencia de knockoff y Lasso se mantiene en 1 cuando k aumenta, mientas que Stepwise decrece ligeramente.

Efecto del parámetro ρ

En lo siguiente la variable ρ toma los valores $\rho = 0.05, 0.15, 0.25, \ldots, 0.85, 0.95$, fijando $A = 50$ y $k = 30$.

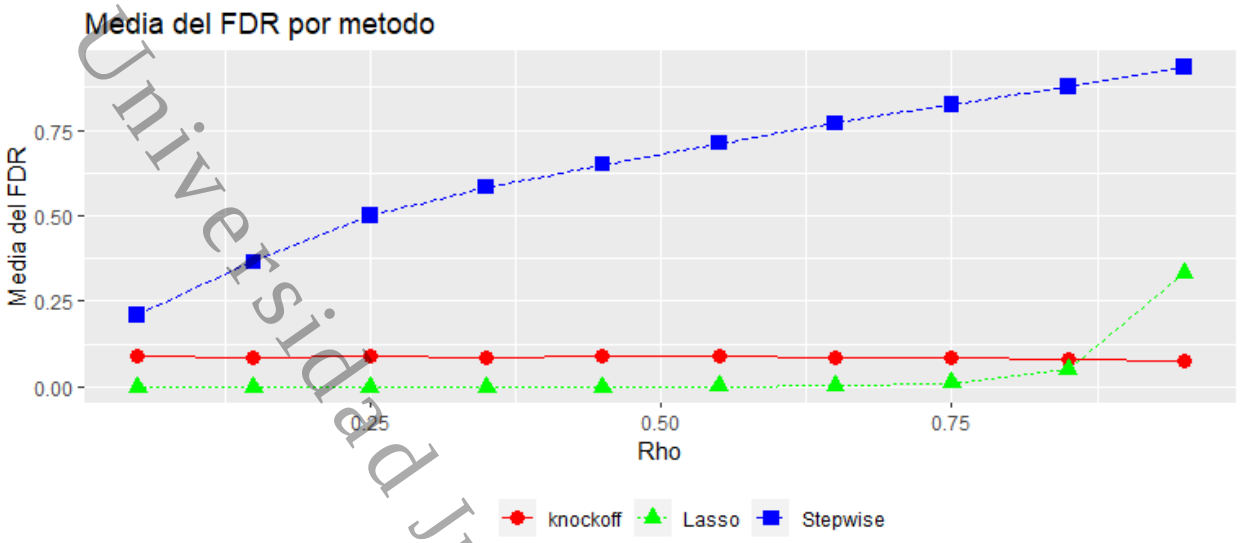

Figura 4.5: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando ρ, con $n = 3000$, $p = 1500$ y $k = 30$, $A = 50$

Observamos que la variación de ρ no afecta significativamente al FDR del método knockoff, a Lasso lo afecta significativamente con $\rho > 0.75$ y a Stepwise notamos que con ρ alta su FDR aumenta. Lasso y knockoff son los que tienen menor FDR.

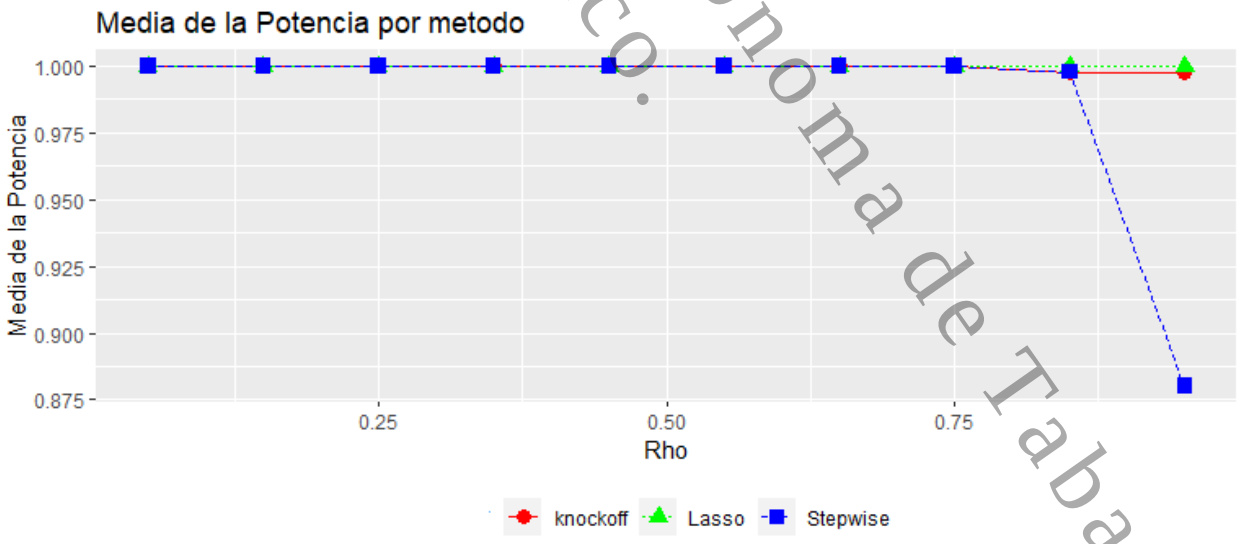

Figura 4.6: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando ρ, con $n = 3000$, $p = 1500$ y $k = 30$, $A = 50$

Por otro lado, la potencia del método knockoff y Lasso no varía significati-

vamente cuando ρ varía, en cambio Stepwise sufre un decremento cuando ρ es cercano a 1. Las potencias de knockoff y Lasso se mantienen cercanas a 1 para los valores de ρ considerados.

4.2.2. Caso II: $p \leq n \leq 2p$

Para el caso $p \leq n \leq 2p$, fijamos $p = 250$ y $n = 250$. Veremos el comportamiento del FDR y la potencia al modificar la amplitud, el número de betas distintas de cero o el parámetro ρ.

Efecto de la amplitud

La amplitud tomará los siguientes valores $A = 10, 20, 30, \ldots, 100$, fijamos $k = 30$ (donde k es el número de betas distintas de cero) y $\rho = 0.3$.

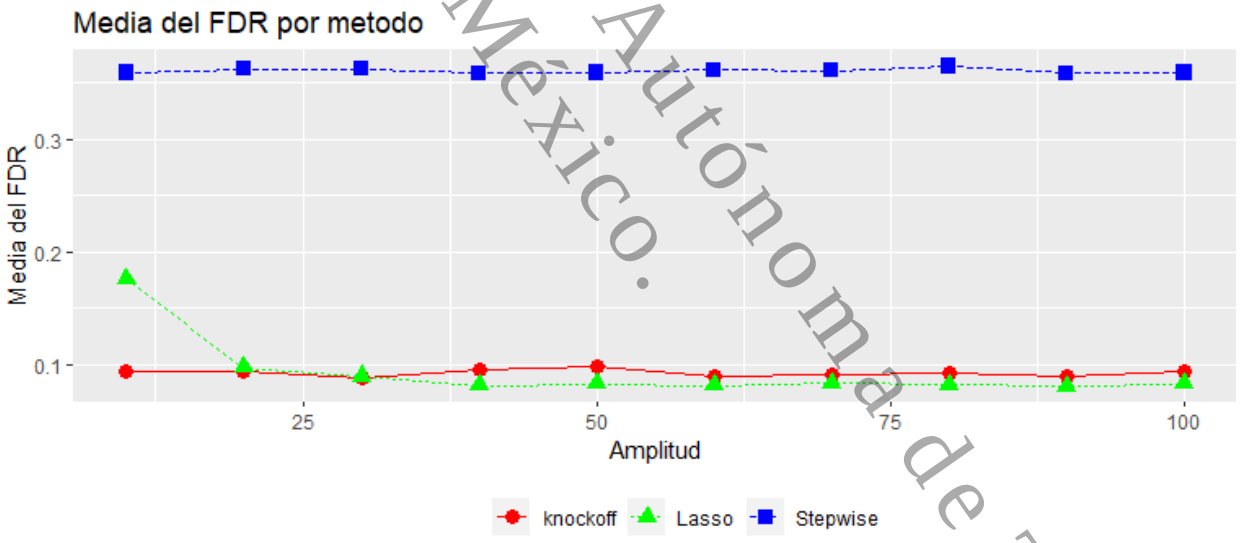

Figura 4.7: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 250$, $p = 250$ y $k = 30$, $\rho = 0.3$

Observamos que para este caso la variación de la amplitud no afecta de forma significativa al FDR de los métodos knockoff y Stepwise, para Lasso notamos que cuando la amplitud crece su FDR decrece. Knockoff y Lasso son los métodos con menor FDR.

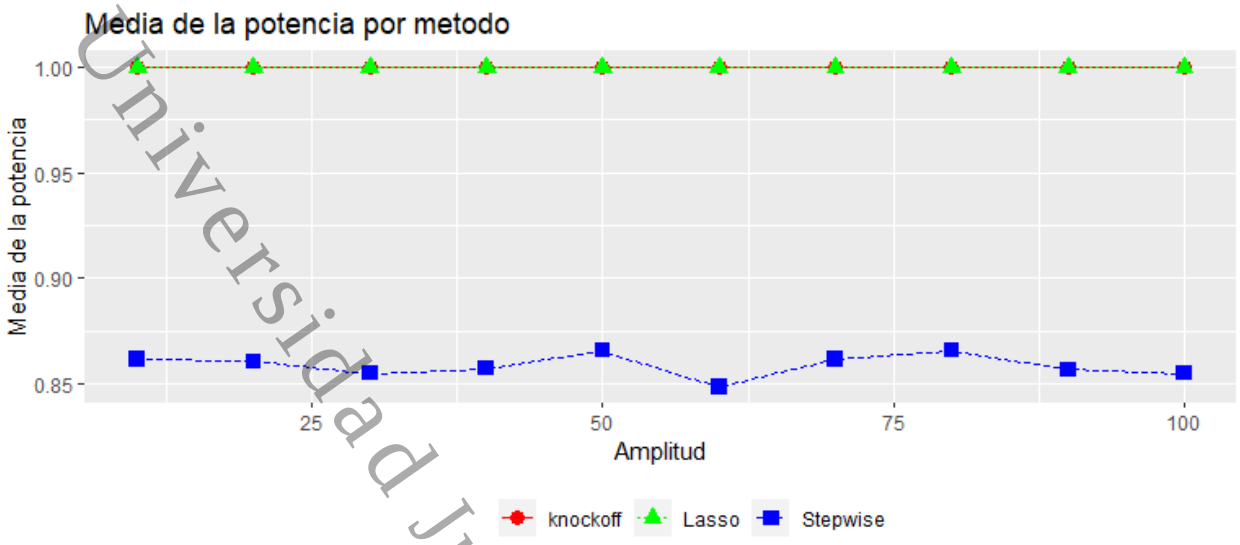

Figura 4.8: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 250$, $p = 250$ y $k = 30$, $\rho = 0.3$

Para la media de la potencia P, notamos que para knockoff y Lasso se mantiene en 1, por otro lado para Stepwise se ve variación, pero se mantiene entre 0.85 y 0.90.

Efecto del aumento de la variable k

El número de betas distintas de cero k tomará los siguientes valores $k = 10, 20, 30, \ldots, 100$, fijando $A = 30$ y $\rho = 0.3$.

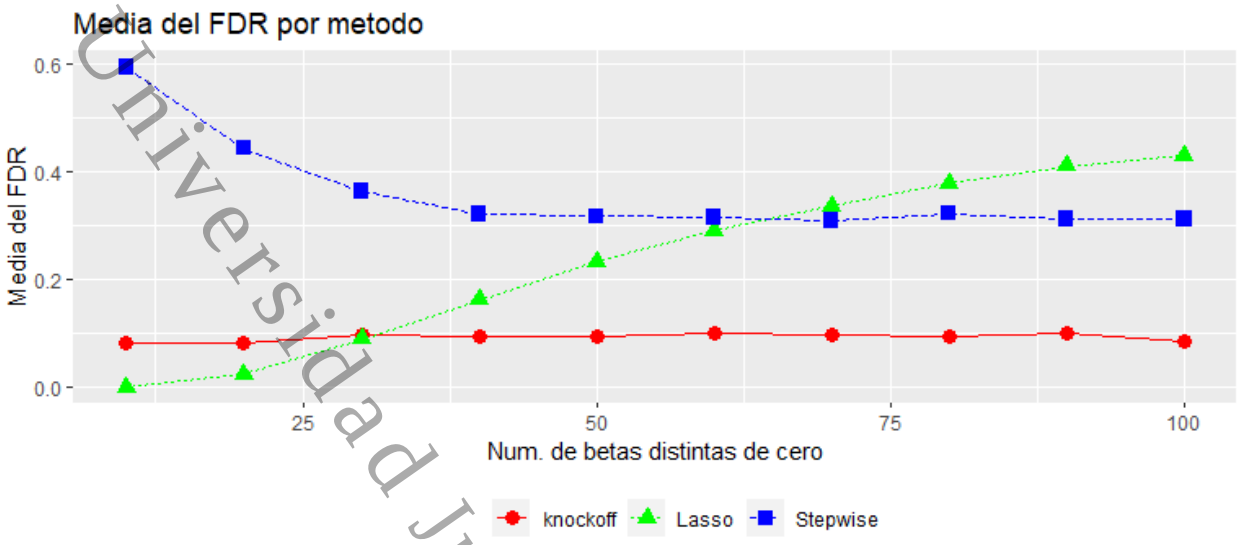

Figura 4.9: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando el número de betas distintas de cero k, con $n = 250$, $p = 250$ y $A = 30$, $\rho = 0.3$

Podemos observar que la variación de k no altera de manera significativa la media del FDR para el método knockoff, en cambio en Lasso notamos que entre más grande es k el FDR aumenta y con Stepwise disminuye. Knockoff es el método que tiene FDR más bajo para casi todos los valores de k considerados.

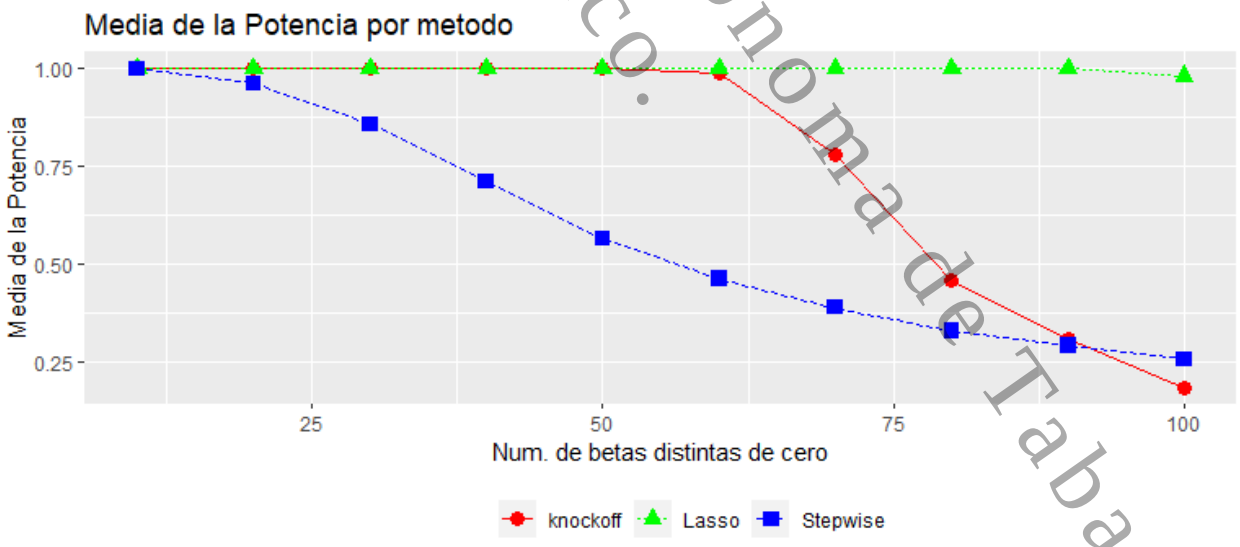

Figura 4.10: Media de la potencia P de los métodos knockoff, Stepwise, Lasso,variando el número de betas distintas de cero k, con $n = 250$, $p = 250$ y $A = 30$, $\rho = 0.3$

Observamos que cuando $k \geq 60$ la media de la potencia P en el método

knockoff decrece, mientras para Lasso no hay una variación significativa y mantiene un valor igual o cercano a 1, mientras que Stepwise decrece cuando k aumenta.

Efecto del parámetro ρ

La variable ρ tomará los valores $\rho = 0.05, 0.15, 0.25, \ldots, 0.85, 0.95$, fijando $A = 30$ y $k = 30$.

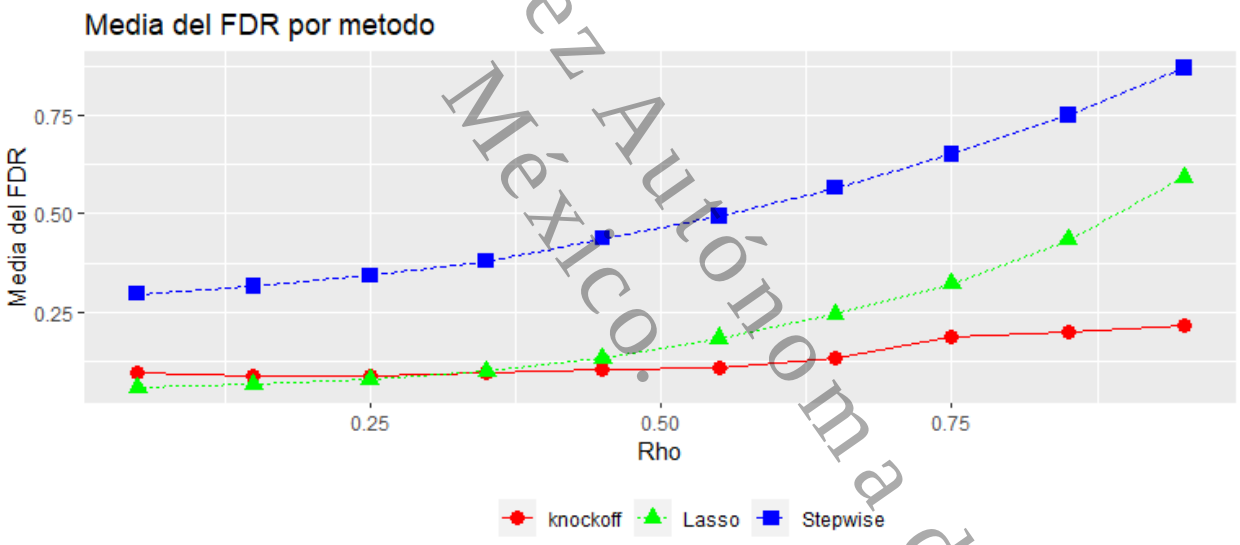

Figura 4.11: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando la ρ, con $n = 250$, $p = 250$ y $k = 30$, $A = 30$

Observamos que la variación de ρ influye en la media del FDR en los tres métodos, para knockoff se ve un ligero incremento, para Lasso crece significativamente y Stepwise aumenta conforme ρ aumenta. Knockoff es el método con menor FDR.

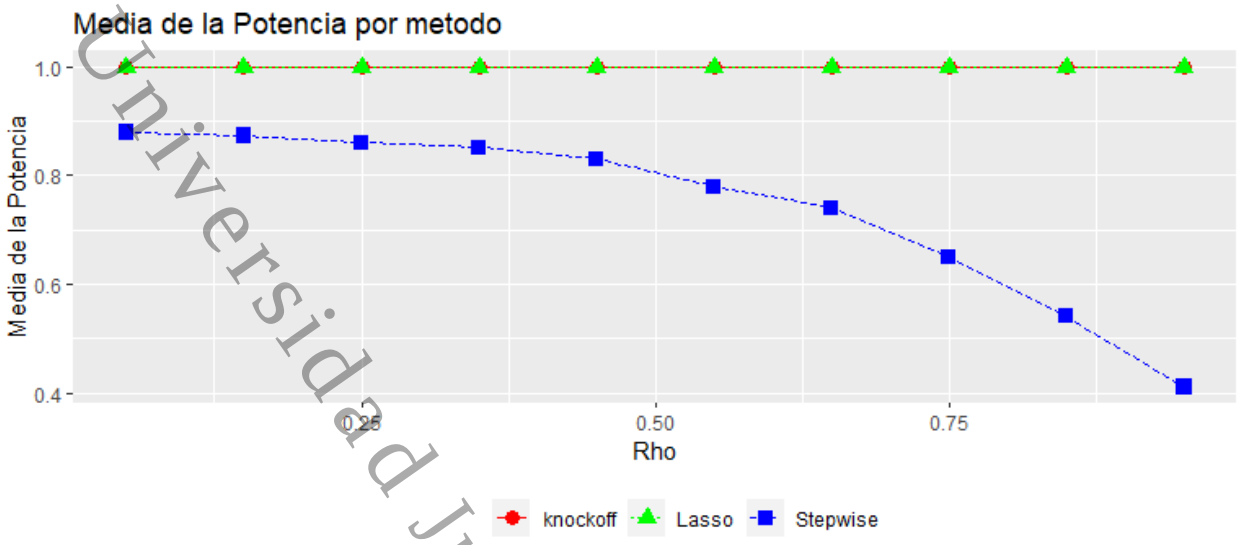

Figura 4.12: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando ρ, con $n = 250$, $p = 250$ y $k = 30$, $A = 30$

Observamos que la variación de ρ no influye de manera significativa en la media de la potencia P para los métodos knockoff y Lasso que se mantiene en 1, por otro lado para Stepwise notamos que mientras ρ crece la potencia disminuye.

4.2.3. Caso III: $n \ll p$

Para el caso $n \ll p$, fijamos $p = 1200$ y $n = 250$. Veremos el comportamiento del FDR y la potencia al modificar la amplitud, el número de betas distintas de cero o el parámetro ρ.

Efecto de la amplitud

La amplitud tomará los valores $A = 10, 20, 30, \ldots, 100$, fijando $k = 30$ (donde k es el número de betas distintas de cero) y $\rho = 0.3$.

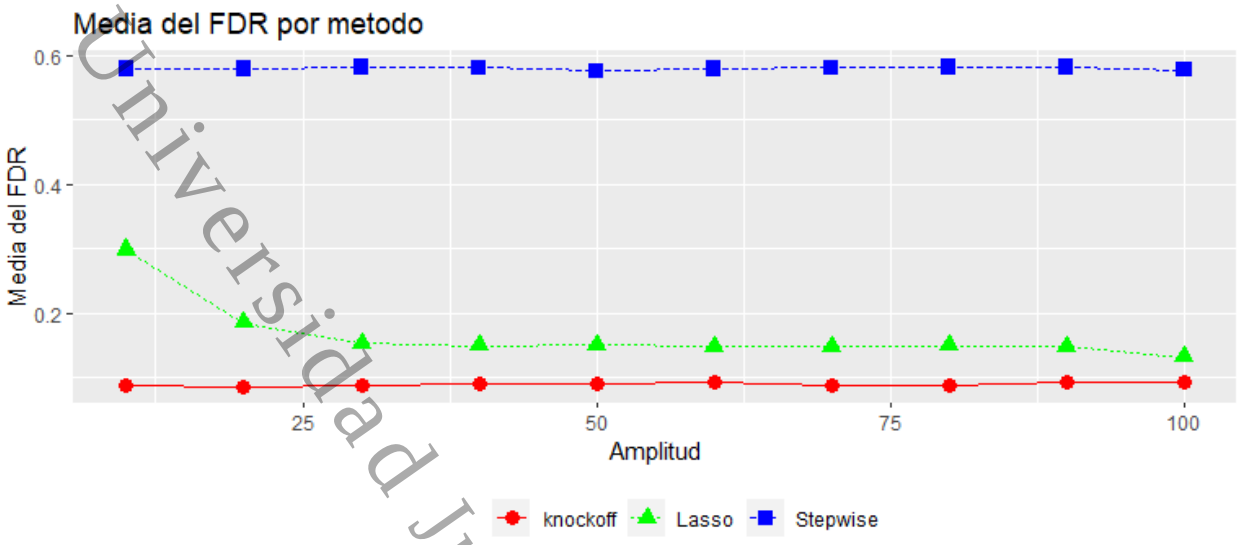

Figura 4.13: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 250$, $p = 1200$ y $k = 30$, $\rho = 0.3$

Observamos que la variación de la amplitud no influye significativamente en la media del FDR en los métodos de knockof y Stepwise, mientras que para Lasso notamos que cuando la amplitud crece el FDR decrece. Los métodos con menor FDR son knockoff y Lasso.

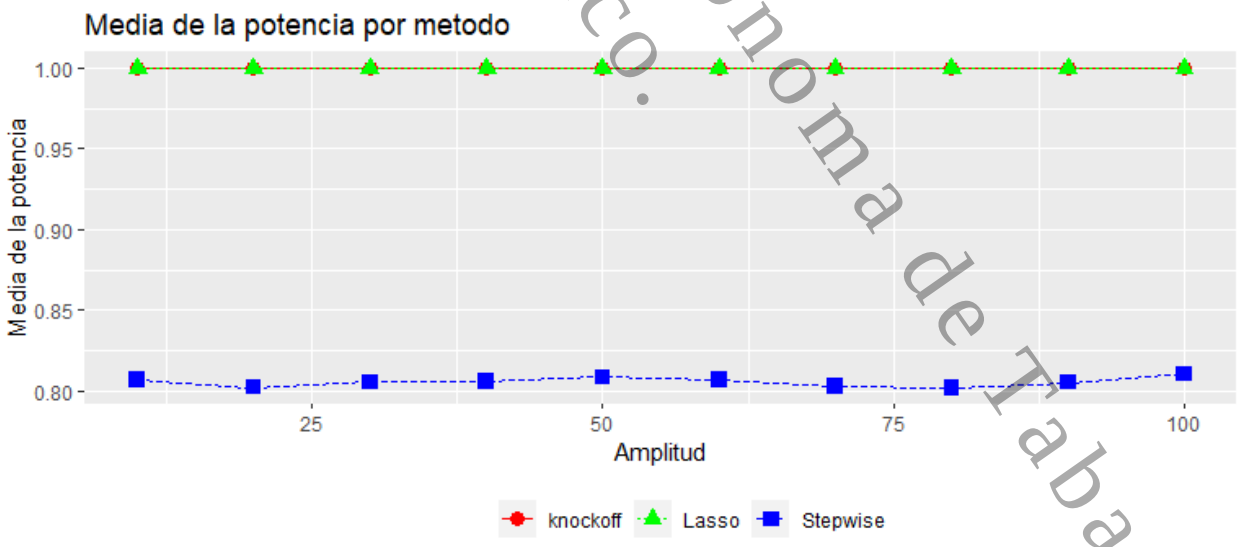

Figura 4.14: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando la amplitud, con $n = 250$, $p = 1200$ y $k = 30$, $\rho = 0.3$

Observamos que la variación de la amplitud no influye significativamente en

la media de la potencia P en los tres modelos. Knockoff y Lasso son los que tienen potencia igual a 1 para los valores considerados de amplitud.

Efecto del aumento de la variable k

En lo siguiente el número de betas distintas de cero k tomará los valores $k = 10, 20, 30, \ldots, 100$, fijando $A = 30$ y $\rho = 0.3$.

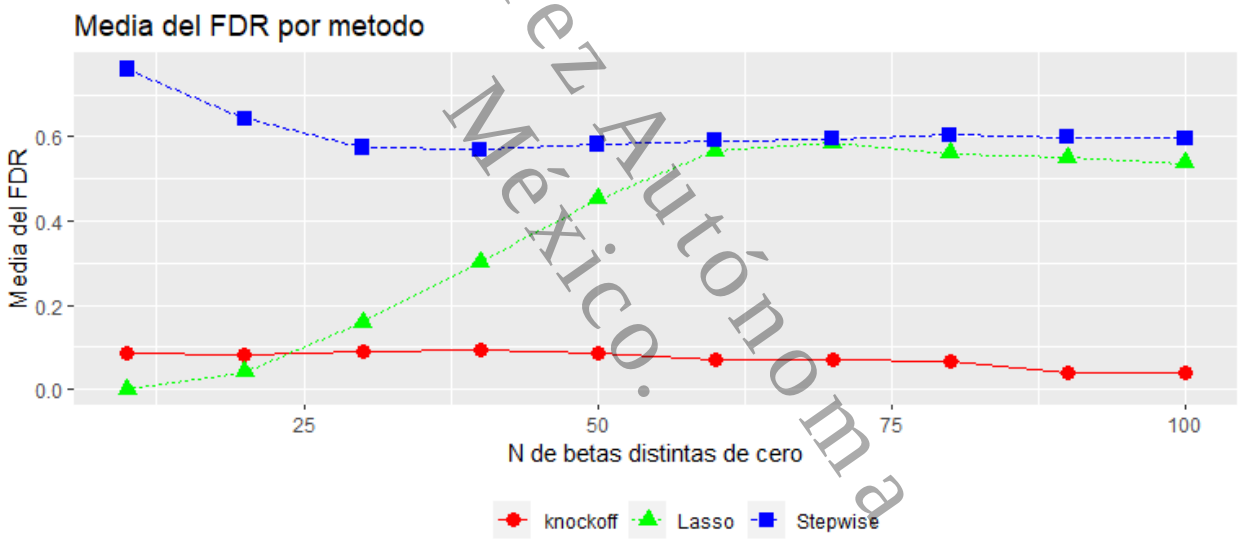

Figura 4.15: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando el número de betas distintas de cero k, con $n = 250$, $p = 1200$ y $A = 30$, $\rho = 0.3$

Observamos que la variación de k no influye significativamente en la media del FDR en knockoff, por otro lado en Lasso podemos ver que aumenta y en Stepwise un ligero decremento. Mayormente knockoff es el método con menor FDR.

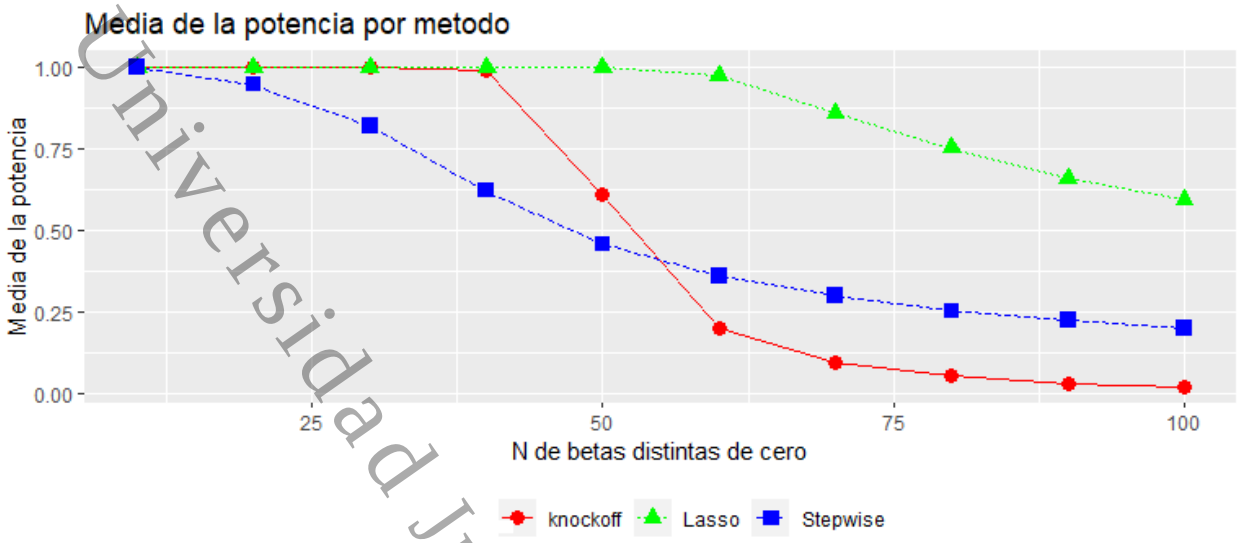

Figura 4.16: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando el número de betas distintas de cero k, con $n = 250$, $p = 1200$ y $A = 30$, $\rho = 0.3$

Por otro lado observamos que la variación de k sí afecta significativamente a la media de la potencia P en los tres métodos, con knockoff vemos que con $k > 40$ la potencia empieza a decrecer significativamente, Lasso con $k > 50$ decrece pero no de manera significativa y Stepwise notamos que decrece conforme k aumenta. Lasso es el método con potencia más alta la mayoría de las veces.

Efecto del parámetro ρ

A continuación la variable ρ tomará los valores $\rho = 0.05, 0.15, 0.25, \ldots, 0.85$, 0.95, fijando $A = 30$ y $k = 30$.

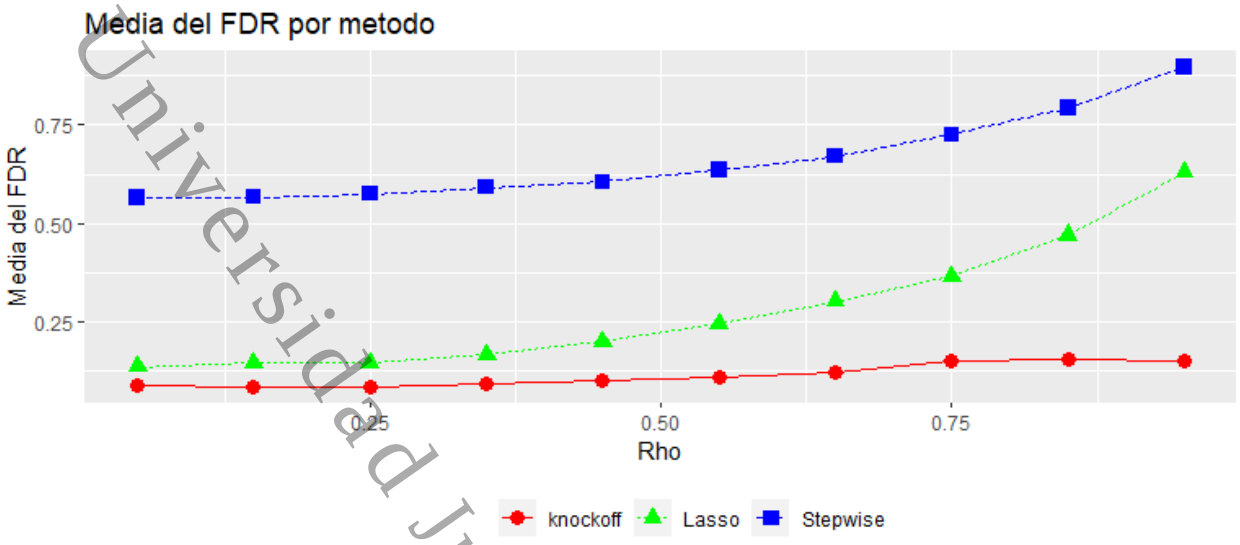

Figura 4.17: Media del FDR de los métodos knockoff, Stepwise, Lasso, variando ρ, con $n = 250$, $p = 1200$ y $k = 30$, $A = 30$

Observamos que la variación de ρ no afecta de manera significativa a la media FDR en el método knockoff, por otro lado notamos que para Lasso y Stepwise si aumenta ρ también aumenta la media del FDR. Knockoff es el método con menor FDR.

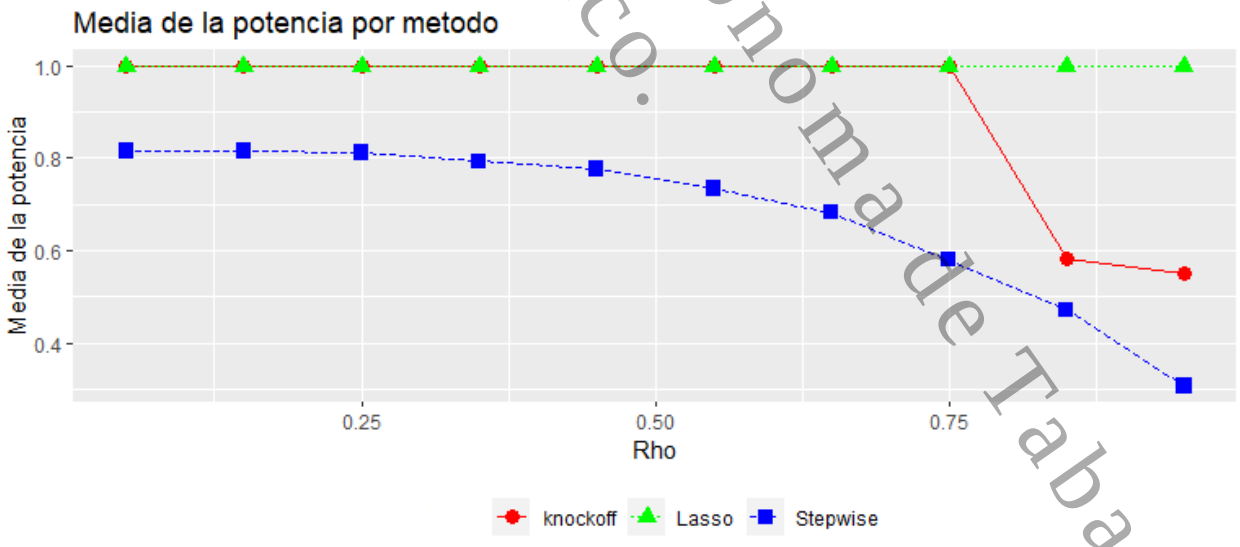

Figura 4.18: Media de la potencia P de los métodos knockoff, Stepwise, Lasso, variando ρ, con $n = 250$, $p = 1200$ y $k = 30$, $A = 30$

Observamos que la media de la potencia P de knockoff disminuye significa-

tivamente cuando $\rho > 0.75$, mientras que con Lasso parece no influir y para Stepwise notamos que si ρ aumenta la media de la potencia decrece. Lasso es el método con potencia igual a 1 en todos los casos considerados.

4.3. Aplicación a datos reales

A continuación aplicaremos el filtro knockoff, el método de BH, la regresión Lasso y el método de Stepwise para detectar mutaciones en el Virus de Inmunodeficiencia Humana Tipo 1 (VIH-1) que están asociadas con la resistencia a los medicamentos. El conjunto de datos, descrito y analizado en [Barber and Candès, 2015], consta de medidas de resistencia a los fármacos e información de genotipo de muestras de VIH-1, con conjuntos de datos separados para la resistencia a los inhibidores de la proteasa (IP), a los inhibidores nucleósidos de la transcriptasa inversa (RT), nucleósidos (NRTI) y a los inhibidores de RT no nucleósidos (NNRTI). En este ejemplo de aplicación agregamos algunos análisis a los que se presentan en [Barber and Candès, 2015].

En cada clase de fármaco, a algunas muestras les faltan medidas de resistencias para alguno de los medicamentos, por lo que para el análisis de cada medicamento el tamaño de muestra y el número de mutaciones presentes serán diferentes para cada una.

Se analizará cada fármaco por separado. La respuesta y_i viene dada por el aumento logarítmico de la resistencia a los fármacos probada en laboratorio en la i-ésima muestra, mientras que el diseño de la matriz X tiene entradas $X_{ij} \in \{0, 1\}$, que indica la presencia o la ausencia de la mutación j en la i-ésima muestra. Para cada fármaco, solo se mantienen esas mutaciones que aparecen tres o más veces en la muestra de ese fármaco y se eliminan las columnas duplicadas de X para permitir la identificación.

Diferentes mutaciones en las mismas posiciones se tratan como características y se asume un valor lineal aditivo sin interacciones. Luego al conjunto de datos resultantes se les aplica el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, también realizamos la regresión Lasso y el método de Stepwise.

Para evaluar los resultados comparamos las mutaciones seleccionadas con los

páneles existentes de mutaciones seleccionadas por tratamientos (TSM) de [Rhee et al., 2005], dado que se trata de un experimento de datos reales, se desconoce la verdad fundamental, pero estos páneles proporciona una buena aproximación que podemos utilizar para evaluar los métodos. Para cada clase de fármaco (IP, NRTI, NNRTI), [Rhee et al., 2005], crean páneles de mutaciones que están presentes con una frecuencia significativamente mayor (después de corregir para comparaciones múltiples) en muestras de virus de individuos que han sido tratados con esas clase de fármacos, en comparación con individuos que nunca han recibido esa clase de fármaco. Por lo tanto, los datos que usamos para la selección del modelo (basados en la resistencia a los medicamentos comprobados en laboratorio) y los páneles de mutación usados para validar nuestros resultados (basados en la asociación con el historial de tratamientos del paciente) provienen de diferentes tipos de estudios y nuestro objetivo es ver la replicabilidad, es decir, evaluaremos los resultados de la selección de nuestros modelos en función de cuantas de las mutaciones identificadas por nuestro análisis aparece también en la lista de TSM. Se sabe que múltiples mutaciones en la misma proteasa o posición de RT a menudo pueden asociarse con resultados relacionados con la resistencia a los medicamentos. Dado que la lista TSM es una aproximación de la verdad fundamental, compararemos solo las posiciones de las mutaciones seleccionadas con las posiciones de las mutaciones en la lista TMS.

4.3.1. Fármacos del tipo PI

APV

Para el fármaco APV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.1: Tasa de falso positivos del fármaco APV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.136	0.32	0.38	0.266

Figura 4.19: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco APV de tipo PI.

Observamos de la figura 4.19 que para los datos de este ejemplo, el método knockoff tiene el menor FDR y una buena cantidad de descubrimientos verdaderos, por otro lado el método de Lasso y Stepwise ofrecen una mayor cantidad de descubrimientos verdaderos pero tiene una mayor FDR.

ATV

Para el fármaco ATV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.2: Tasa de falso positivos del fármaco ATV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.29	0.259	0.315	0.086

Figura 4.20: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco ATV de tipo PI.

Observamos de la figura 4.20 que para los datos de este ejemplo, el método de Stepwise sería una buena elección ya que ofrece un menor FDR y una buena cantidad de descubrimientos verdaderos y el método knockoff tiene un alto FDR y una buena cantidad de descubrimientos verdaderos.

IDV

Para el fármaco IDV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.3: Tasa de falso positivos del fármaco IDV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.387	0.333	0.431	0.333

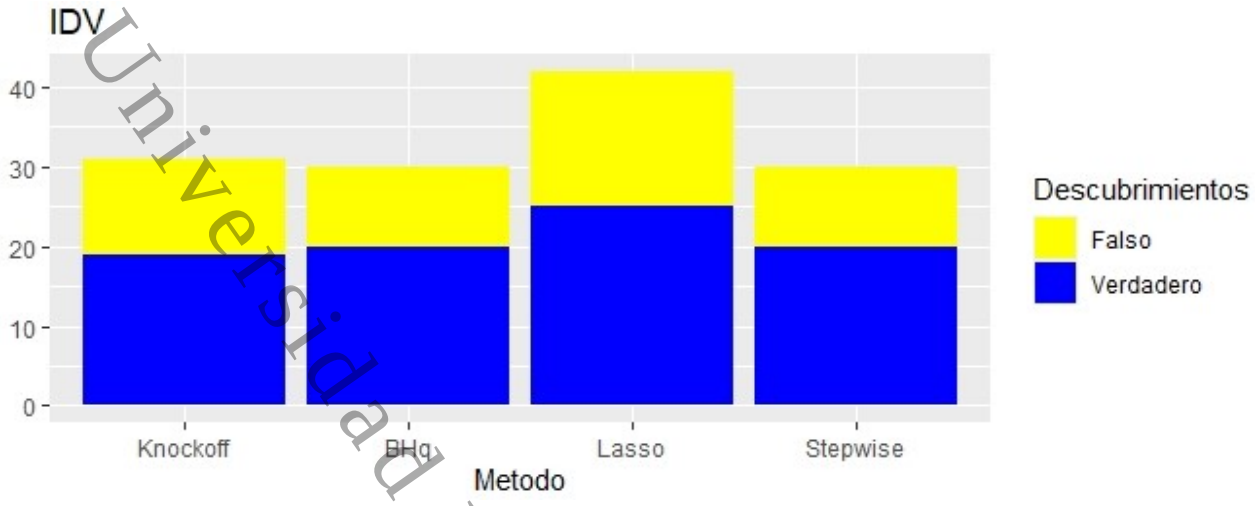

Figura 4.21: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco IDV de tipo PI.

Observamos de la figura 4.21 que el filtro knockoff tiene un FDR un poco mayor a los métodos BHq y Stepwise, mientras que Lasso tiene el FDR más alto, pero tiene más variables verdaderas.

LPV

Para el fármaco LPV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.4: Tasa de falso positivos del fármaco LPV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.058	0.142	0.3	0.28

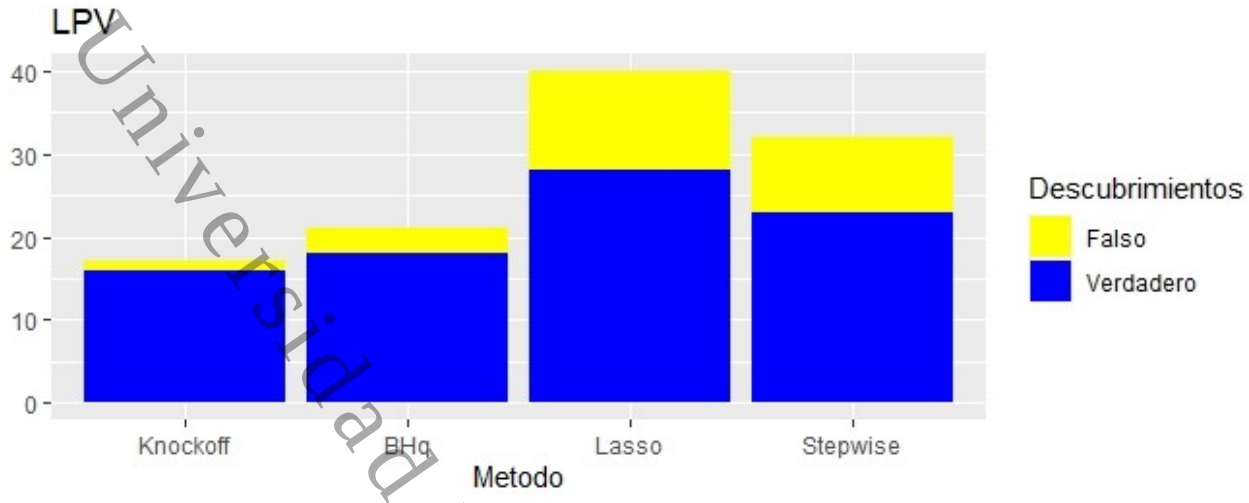

Figura 4.22: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco LPV de tipo PI.

Observamos de la figura 4.22 que el filtro de knockoff tiene la menor FDR, pero menos descubrimientos verdaderos, los métodos BHq, Stepwise y Lasso proporcionan mayores descubrimientos verdaderos pero tienen una mayor FDR.

NFV

Para el fármaco NFV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.5: Tasa de falso positivos del fármaco NFV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.225	0.241	0.437	0.258

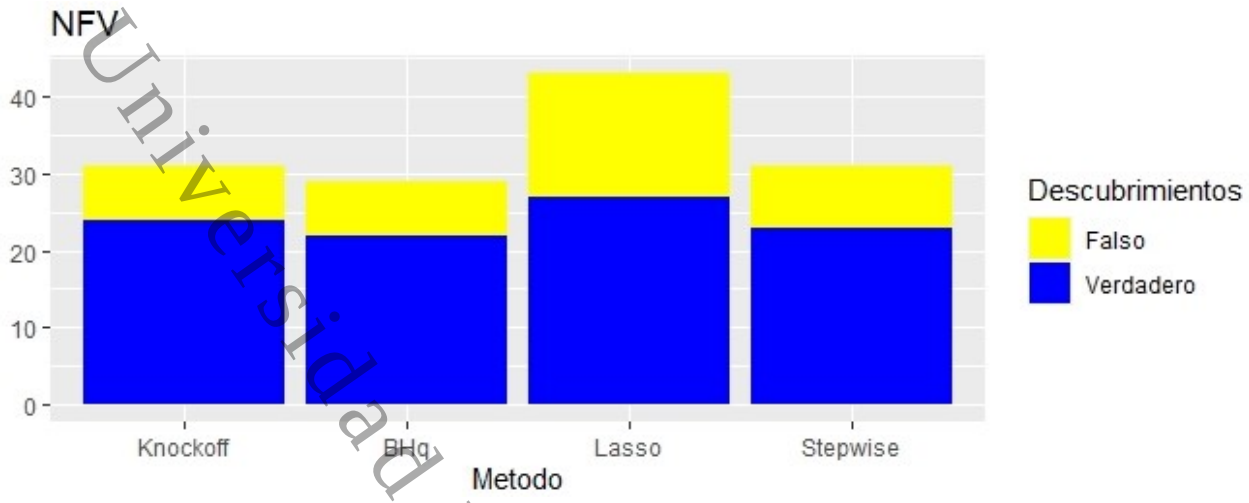

Figura 4.23: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco NFV de tipo PI.

Observamos de la figura 4.23 que el FDR del filtro knockoff es ligeramente más bajo que el de los métodos BHq y Stepwise y además tienen más descubrimientos verdaderos que estos dos métodos, para el método de Lasso observamos que su FDR es el mayor pero también es el que tiene mayor número de descubrimientos verdaderos.

RTV

Para el fármaco RTV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.6: Tasa de falso positivos del fármaco RTV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.296	0.307	0.325	0.225

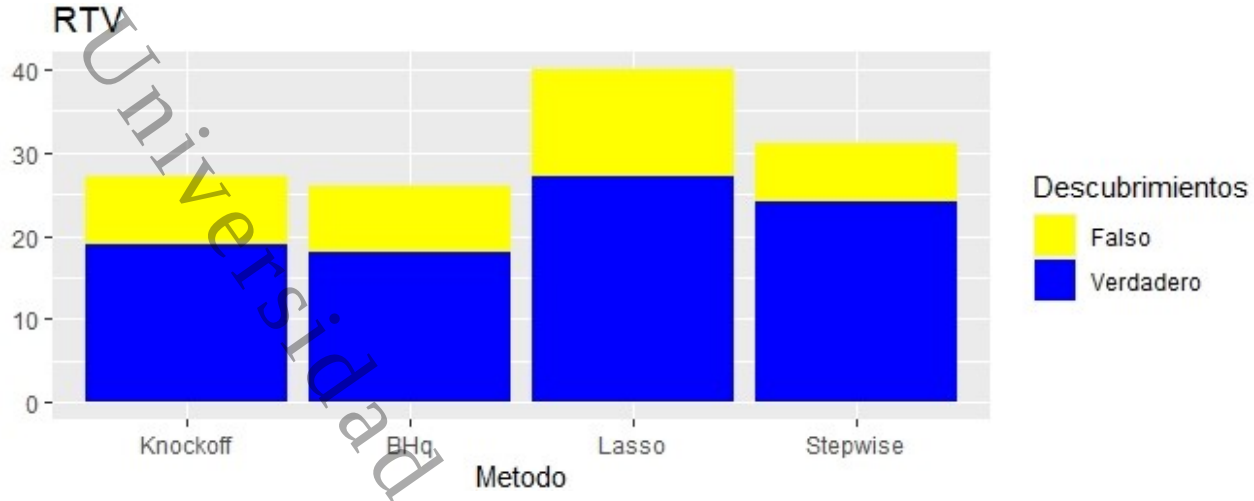

Figura 4.24: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco RTV de tipo PI.

Observamos de la figura 4.24 que en este ejemplo el método de Stepwise es una buena opción, ya que tiene el menor FDR y es el segundo mejor en descubrimientos verdaderos.

SQV

Para el fármaco SQV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.7: Tasa de falso positivos del fármaco SQV de la clase PI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.19	0.366	0.333	0.214

Figura 4.25: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco APV de tipo PI.

Observamos en la figura 4.25 que el filtro knockoff tiene el FDR más bajo, aunque es el de menos descubrimientos verdaderos, una buena opción en este caso es el método Stepwise, ya que es el segundo en menos descubrimientos falsos y también el segundo en más descubrimientos verdaderos.

4.3.2. Fármacos del tipo NRTI

X3CT

Para el fármaco X3CT se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.8: Tasa de falso positivos del fármaco X3CT de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0	0	0.533	0.333

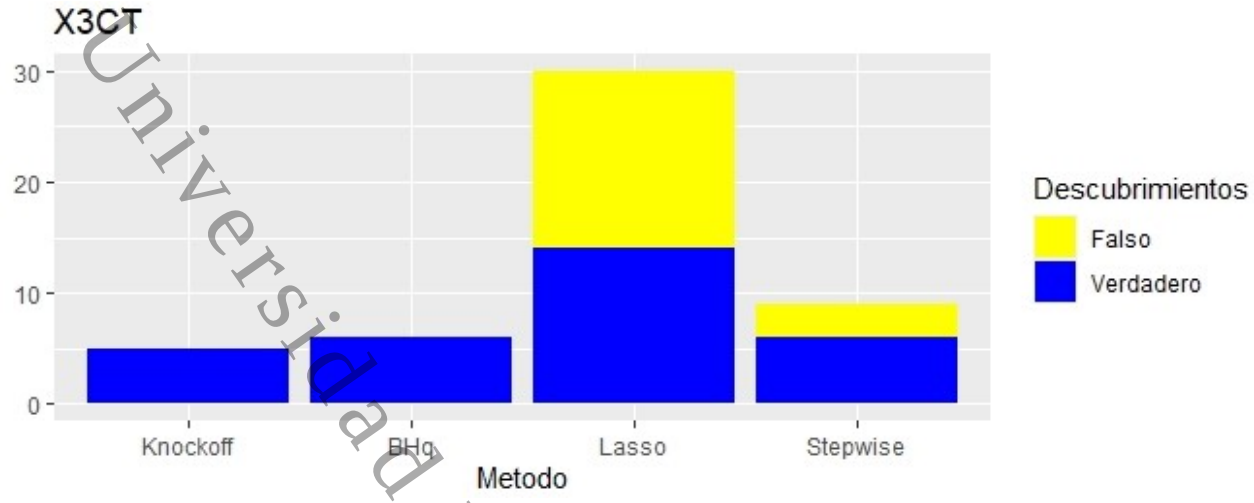

Figura 4.26: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco X3CT de tipo NRTI.

Observamos en la figura 4.26 que el filtro de knockoff no tiene descubrimientos falsos, pero tiene el menor número de descubrimientos verdaderos, una buena opción para este ejemplo sería el BHq ya que también no tiene descubrimientos falsos, pero tiene más número de descubrimientos verdaderos que el filtro de knockoff.

TDF

Para el fármaco TDF se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.9: Tasa de falso positivos del fármaco TDF de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.2	0.555	0.2	0.272

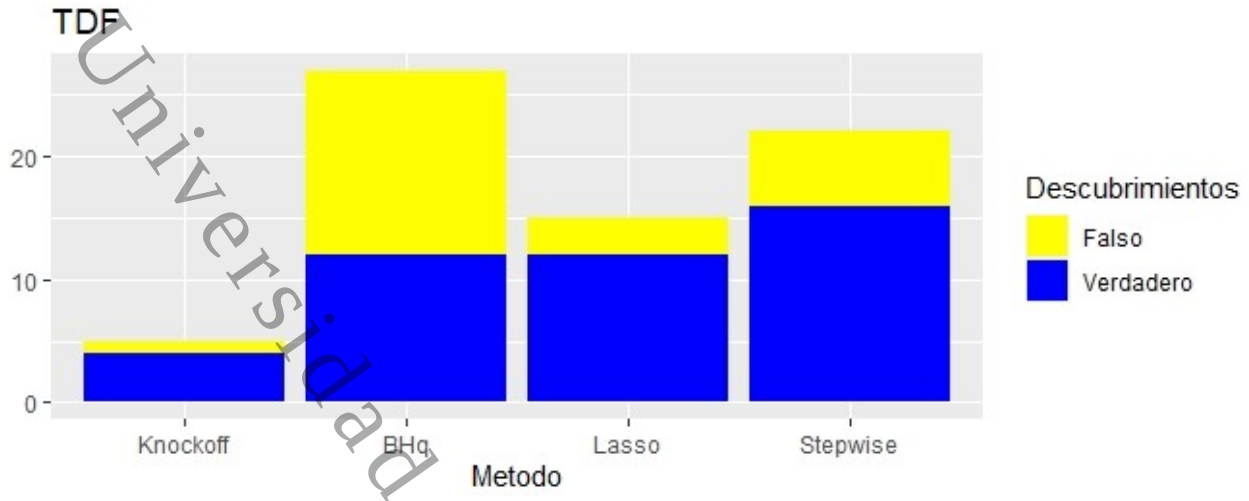

Figura 4.27: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco TDF de tipo NRTI.

Observamos en la figura 4.27 que en este ejemplo el filtro knockoff tiene un FDR bajo, pero un bajo número de descubrimientos falsos, en este caso el método Lasso es una mejor opción ya que tiene un más número de descubrimientos verdaderos y es el segundo en menos descubrimientos falsos.

DDI

Para el fármaco DDI se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.10: Tasa de falso positivos del fármaco DDI de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.166	0.333	0.4	0375

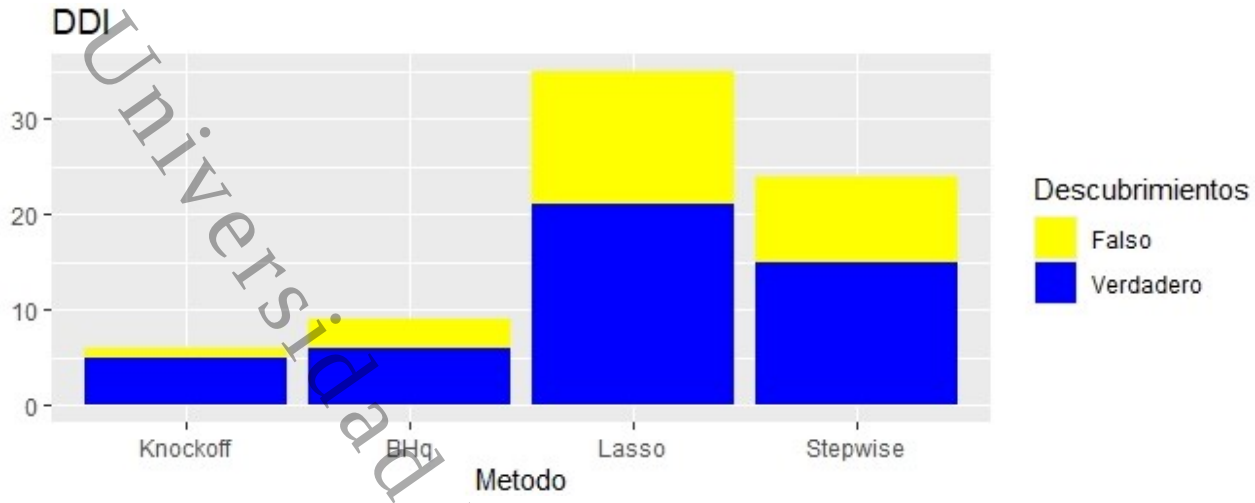

Figura 4.28: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco DDI de tipo NRTI.

Observamos en la figura 4.28 que en este ejemplo el filtro knockoff es el de menor FDR, pero también ofrece muy pocos descubrimientos verdaderos, al igual que el método BHq.

D4T

Para el fármaco D4T se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.11: Tasa de falso positivos del fármaco D4T de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.645	0.5	0.538	0.285

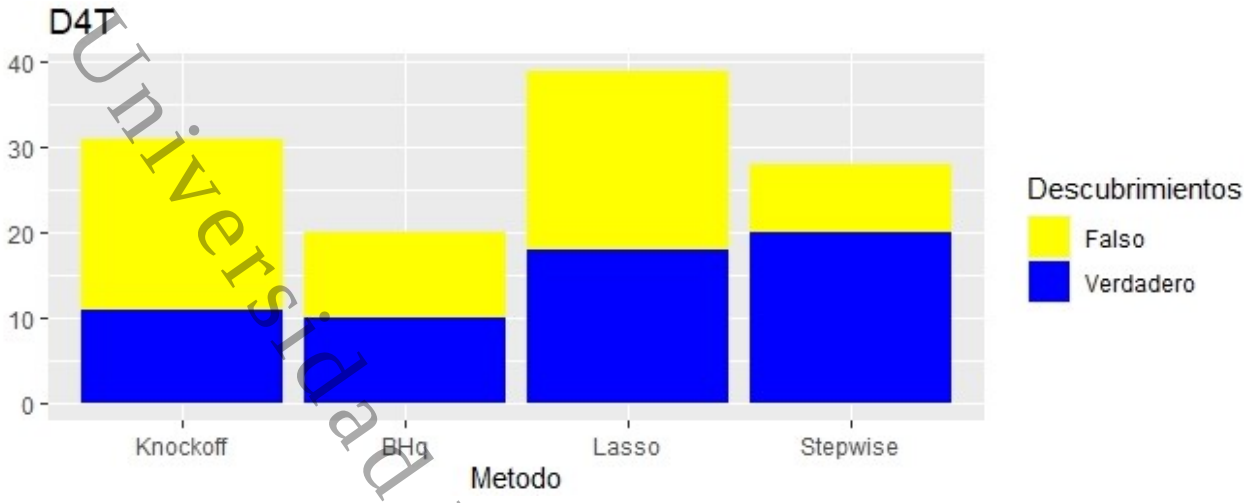

Figura 4.29: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco D4T de tipo NRTI.

Observamos en la figura 4.29 que en este ejemplo la mejor opción puede ser el método Stepwise, ya que ofrece menor FDR y un buen número de descubrimiento verdaderos, podemos notar que en este caso el filtro knockoff tiene el FDR más alto.

AZT

Para el fármaco AZT se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.12: Tasa de falso positivos del fármaco AZT de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0	0.409	0.607	0.366

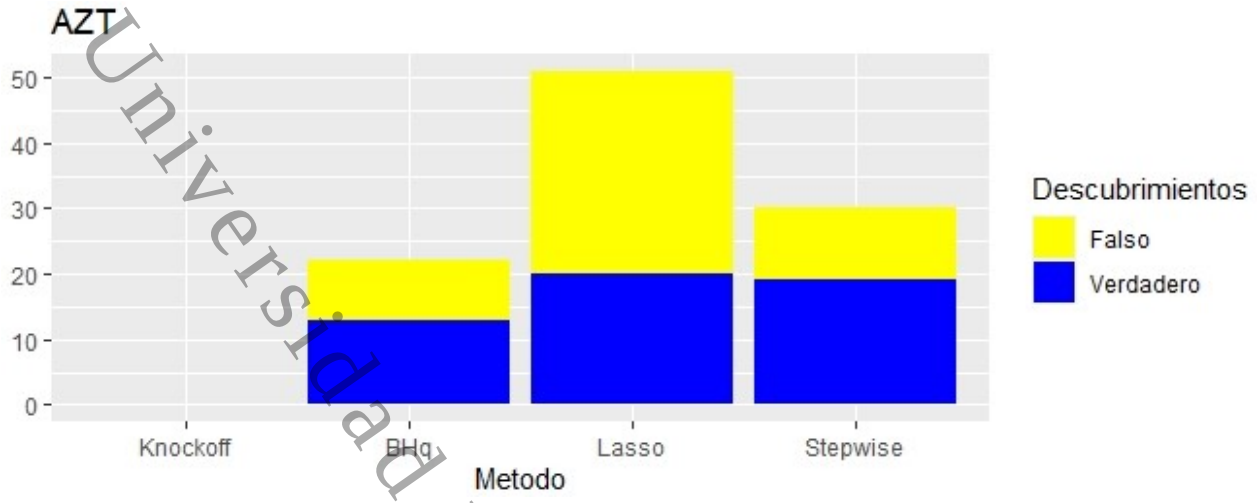

Figura 4.30: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco AZT de tipo NRTI.

Observamos en la figura 4.30 que en este ejemplo el filtro knockoff no seleccionó ninguna variable, y en este caso, una buena opción sería el método Stepwise, ya que tiene la menor tasa de descubrimientos falsos y la segunda mayor tasa de descubrimientos verdaderos.

Repetimos la selección para el fármaco AZT, pero utilizando $q = 0.5$, y se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.13: Tasa de falso positivos del fármaco AZT de la clase NRTI utilizando $q = 0.5$ por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.65	0.586	0.607	0.366

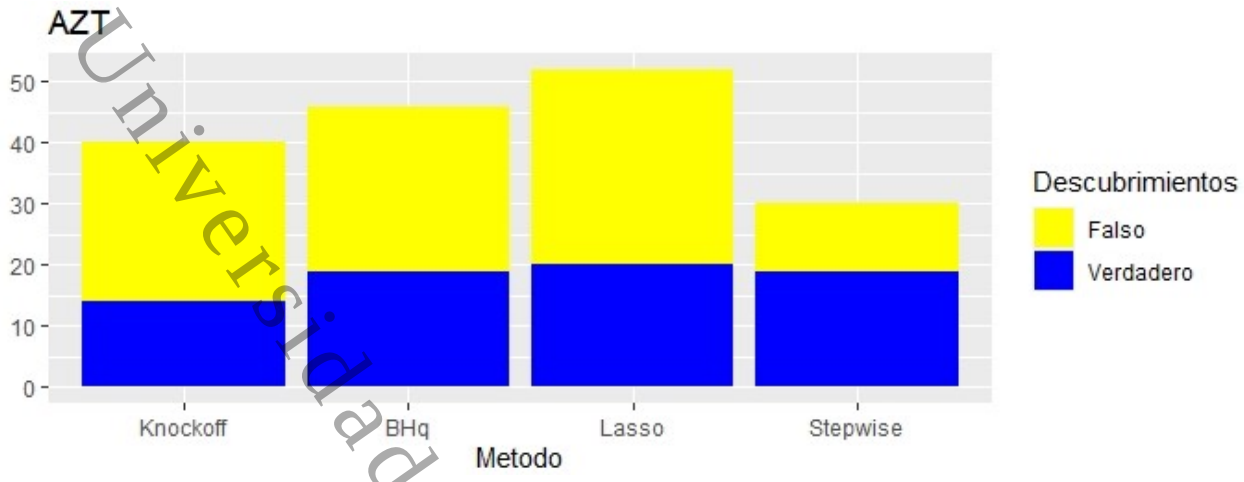

Figura 4.31: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .5$, la regresión Lasso y el método de Stepwise al fármaco AZT de tipo NRTI.

Observamos en la figura 4.31 que en este ejemplo el filtro knockoff ya selecciona variables, pero tiene la mayor FDR, y nuevamente en este caso una buena opción sería el método Stepwise.

ABC

Para el fármaco ABC se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.14: Tasa de falso positivos del fármaco ABC de la clase NRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.166	0.4	0.55	0.32

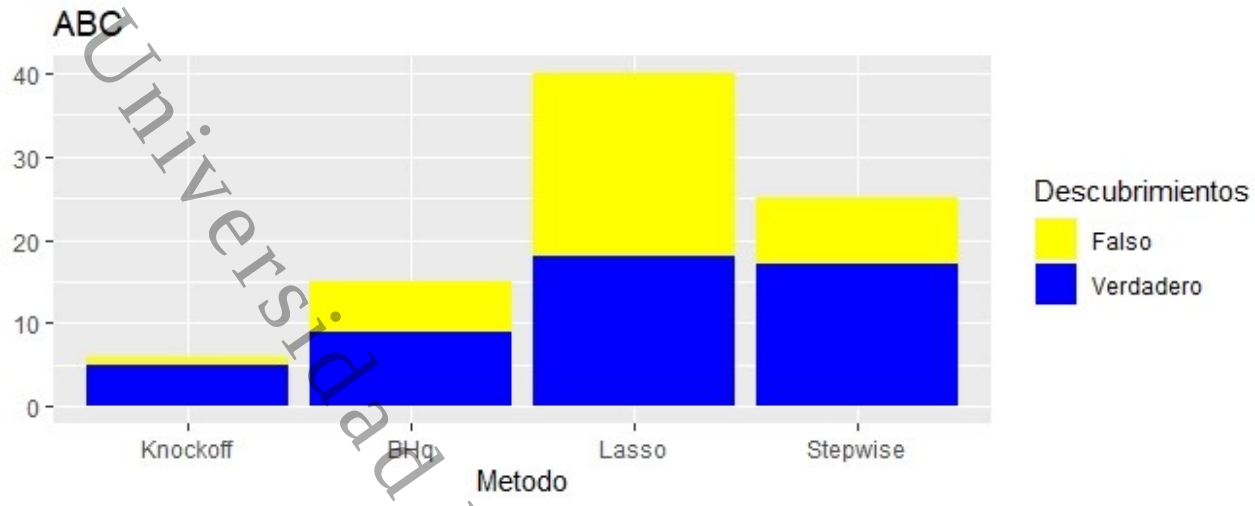

Figura 4.32: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco ABC de tipo NRTI.

Observamos en la figura 4.32 que en este ejemplo el filtro knockoff tiene la menor FDR, pero una baja selección de variables verdaderas y en este caso una buena opción sería el método Stepwise.

4.3.3. Fármacos del tipo NNRTI

DLV

Para el fármaco DLV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.15: Tasa de falso positivos del fármaco DLV de la clase NNRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.583	0.628	0.769	0.588

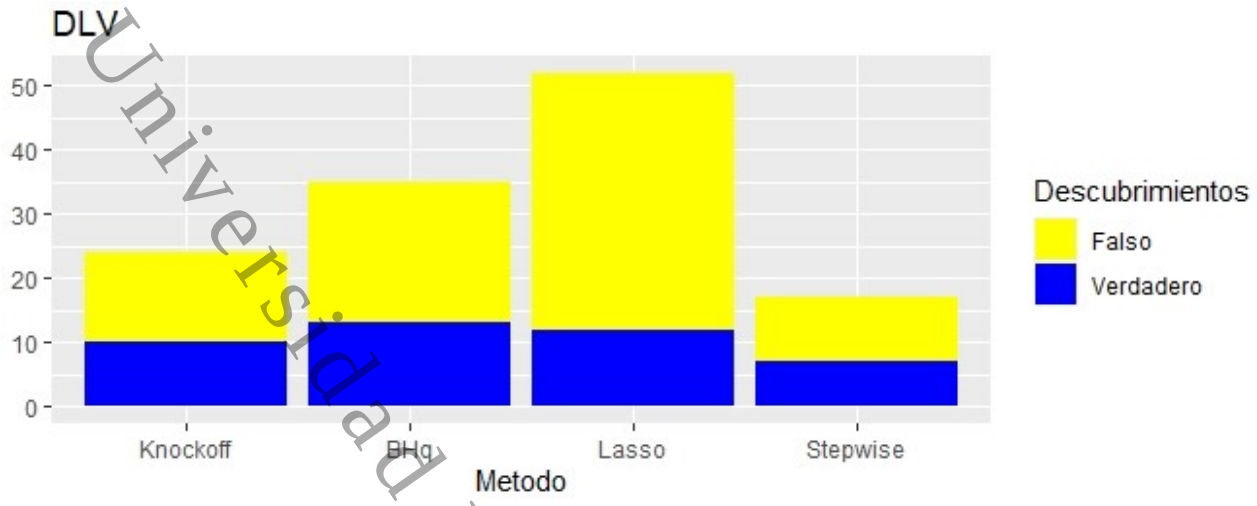

Figura 4.33: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco DLV de tipo NNRTI.

Observamos en la figura 4.33 que en este ejemplo todo los métodos obtienen un alto FDR, al igual que bajos descubrimientos verdaderos.

EFV

Para el fármaco EFV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.16: Tasa de falso positivos del fármaco EFV de la clase NNRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.5	0.473	0.641	0.625

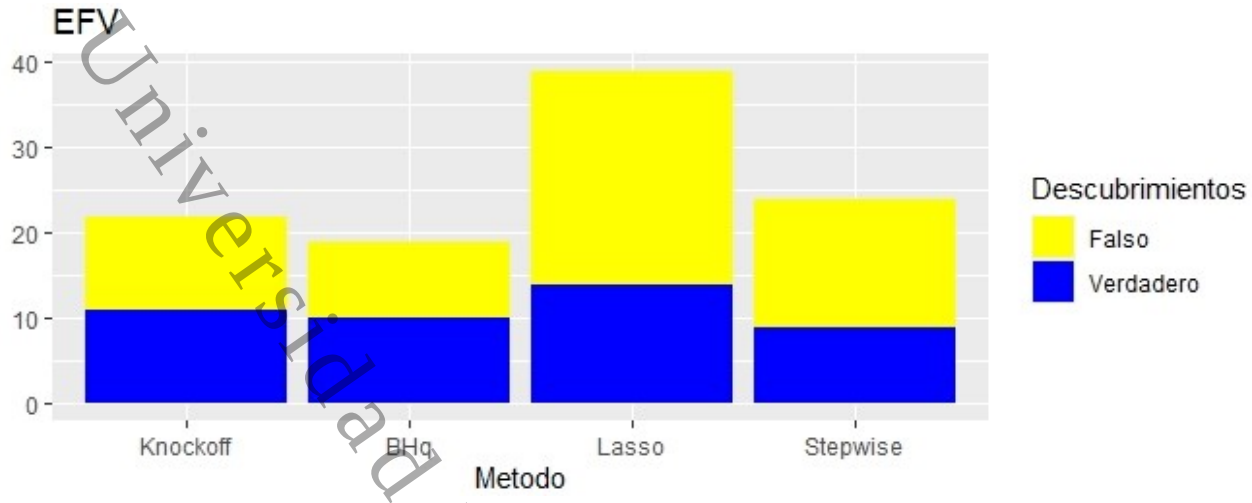

Figura 4.34: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco EFV de tipo NNRTI.

Observamos en la figura 4.34 que en este ejemplo todo los métodos obtienen un alto FDR, al igual que un similar número de descubrimientos verdaderos.

IDV

Para el fármaco IDV se obtuvieron las siguientes tasas de falsos positivos:

Tabla 4.17: Tasa de falso positivos del fármaco IDV de la clase NNRTI por método.

Método	knockoff	BHq	Lasso	Stepwise
FDR	0.588	0.636	0.745	0.526

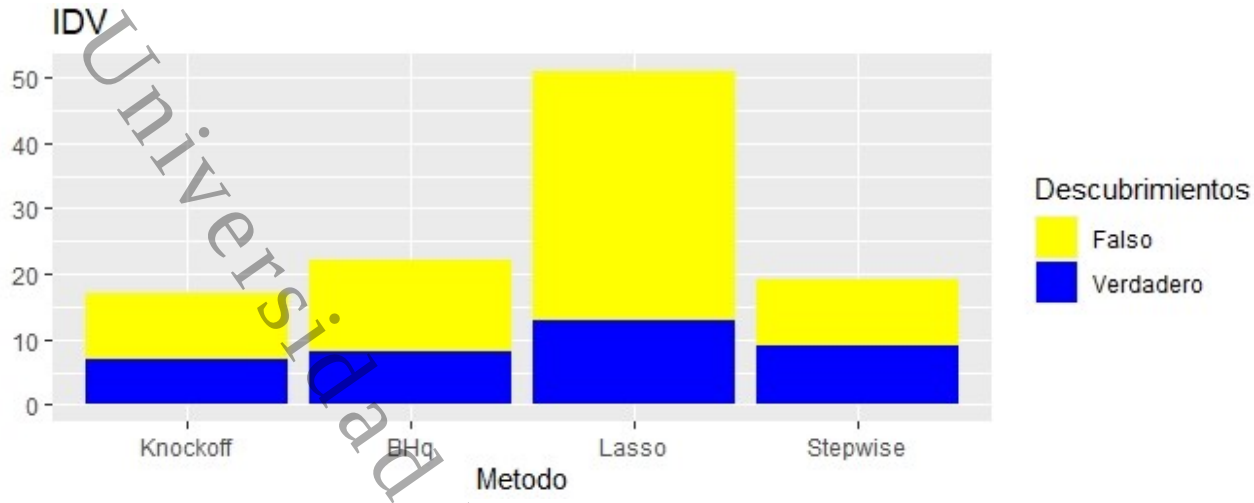

Figura 4.35: Resultado de aplicar el filtro knockoff y el algoritmo de BHq, cada uno con $q = .2$, la regresión Lasso y el método de Stepwise al fármaco IDV de tipo NNRTI.

Observamos en la figura 4.35 que en este ejemplo todo los métodos obtienen un alto FDR, al igual que un similar número de descubrimientos verdaderos.

Conclusiones

En este trabajo se estudiaron los métodos de selección de variables knockoff, Lasso y Stepwise. Estos métodos se aplicaron a regresiones lineales simuladas para observar sus tasas de falsos positivos y potencias, variando diferentes parámetros para analizar su eficiencia.

En las simulaciones se observó que en el caso donde $p < n$ las metodologías Lasso y knockoff produjeron las menores tasas de falsos positivos, siendo Lasso la que generalmente tuvo las tasas más bajas, además de tener una potencia igual o cercana a uno, es decir, seleccionó bien a las variables importantes. En los casos $p \leq n \leq 2p$ y $n \ll p$ se observó que generalmente la metodología knockoff produjo las menores tasas de falsos positivos, sin embargo Lasso fue la que generalmente tuvo potencia igual o cercana a uno.

En la aplicación a datos reales, se observó que en la mayoría de los casos la metodología knockoff tuvo la menor tasa de descubrimientos falsos, y en los casos en los que no, notamos que no hubo ningún método en particular que siempre tuviera el menor FDR. Sin embargo, notamos que mayormente knockoff es el que tiene la tasa de descubrimientos verdaderos más pequeña, siendo Lasso el método con esta tasa más alta en la mayoría de los casos.

Apéndice A

Algunos detalles técnicos

A continuación se presentan algunos resultados utilizados en la tesis y detalles técnicos.

A.1. Algunos resultados de la distribución uniforme

- Si $U_1, \ldots, U_N$ son i.i.d. $U(0,1)$ y si $X = \#$ de $U_i's \leq t$, entonces $X \sim Bin(N,t)$.
- Supongamos s y t tales que $0 < s < t \leq 1$. Sea $U \sim U(0,1)$, entonces

$$P(U \leq s \mid U \leq t) = \frac{P(U \leq s)}{P(U \leq t)} = \frac{s}{t}.$$

- Si $U_1, \ldots, U_N$ son i.i.d. $U(0,1)$ y

$$X = \#\{U_i \leq s\},$$
$$Y = \#\{U_i \leq t\},$$

entonces $X \mid Y$ es una variable aleatoria que cuenta cuantas $U_i's$ son menores que s, dado que ya sabemos que hay Y menores que t. Entonces

$$X \mid Y \sim Bin\left(Y, \frac{s}{t}\right).$$

A.2. Teorema de paro óptimo para martingalas

Teorema A.2.1 (Teorema de paro óptimo) *Sea $(\mathcal{F}_t)_{t\geq 0}$ una filtración definida sobre un espacio de probabilidad $(\Omega, \mathcal{F}, \mathbb{P})$ y sea $(M_t)_{t\geq 0}$ un proceso estocástico adaptado a la filtración $(\mathcal{F}_t)_{t\geq 0}$, cuyos caminos son continuos por la derecha y localmente acotados. Las siguientes propiedades son equivalentes:*

- $(M_t)_{t\geq 0}$ *es una martingala con respecto a la filtración* $(\mathcal{F}_t)_{t\geq 0}$.
- *Para cualquier tiempo de paro acotado casi seguramente de la filtración* $(\mathcal{F}_t)_{t\geq 0}$ *tal que* $E(\mid M_T \mid) < +\infty$, *tenemos que* $E(M_T) = E(M_0)$.

A.3. Demostración de $Q(t_q) = q$

Sea $p_{(0)} = 0$. Por la forma general de $Q(t)$, tenemos que si $q \in (0,1)$ y $j = i_{\text{máx}}$ es el índice más grande tal que $Q(p_{(1)}) \leq q$ (o bien $p_{(i)} \leq \frac{i}{N}q$), entonces se cumple:

Caso 1: Para $j = 0$

Se tiene que $Q(t) = Nt$, $\forall t \in [0, p_{(1)})$, y tenemos que $0 < q < Np_{(1)}$, entonces por el teorema del valor intermedio, existe $x \in [0, p_{(1)})$ tal que $Q(x) = q$. Es fácil ver que x cumple con ser la mínima cuota superior de $\{t : Q(t) \leq q\}$, por lo que $x = t_q$, así $Q(t_q) = q$.

Caso 2: $j \in \{1, 2, \ldots, N-1\}$

En este caso tenemos que $Q(t) = \frac{Nt}{j}$, $\forall t \in [p_{(j)}, p_{(j+1)})$ y tenemos que

$$Q(p_{(j)}) = \frac{Np_{(j)}}{j} \leq q < \frac{Np_{(j+1)}}{j+1} < \frac{Np_{(j+1)}}{j},$$

entonces por el teorema del valor intermedio, existe $x \in [p_{(j)}, p_{(j+1)})$ tal que $Q(x) = q$. Es fácil ver que x cumple con ser la mínima cuota superior de $\{t : Q(t) \leq q\}$, por lo que $x = t_q$, así $Q(t_q) = q$.

Caso 3: $j = N$

En este caso se tiene que $Q(t) = t$, $\forall t \in [p_{(N)}, 1]$ y tenemos que

$$Q(p_{(N)}) = p_{(N)} \leq q < 1,$$

entonces por el teorema del valor intermedio, existe $x \in [p_{(N)}, 1)$ tal que $Q(x) = q$. Es fácil ver que x cumple con ser la mínima cuota superior de $\{t : Q(t) \leq q\}$, por lo que $x = t_q$, así $Q(t_q) = q$.

Apéndice B

Códigos

En el siguiente apéndice se mostraran los códigos utilizados para realizar las simulaciones de los ejemplos presentados en los capítulos anteriores.

Simulaciones de Regresión lineal

Regresión lineal en dimensiona baja

```
library(glmnet)
library(knockoff)
library(dplyr)
library(readxl)
library(data.table)
library(bigstep)
library(openxlsx)
library(ggplot2)
library(openxlsx)
library(readxl)
library(doMC)
library(multtest)
```

Listing B.1: Librerías

```
set.seed(4567)

n1=200        # Número de observaciones.
p1=100        # Número de variables.
k1=30         # Número de variables con coeficientes distintos de cero.

# Generar las variables a partir de una distribución normal multivariada.

mu1 = rep(0,p1)
rho1 = 0.3
Sigma1 = toeplitz(rho1^(0:(p1-1)))
X1 = matrix(rnorm(n1*p1),n1) %*% chol(Sigma1)

# Generate the response from a linear model
```

```
15
16 nonzero1 = sample(p1, k1)
17 beta1 = 5 * (1:p1 %in% nonzero1)
18 Y1 = X1 %*% beta1 + rnorm(n1)
```

Listing B.2: Creación de la regresión lineal.

```
1 fdp = function(selected) sum(beta[selected] == 0)/max(1,length(selected))
```

Listing B.3: Calculo del FDR

```
1 d = prepare_data(Y1, X1)
2 fs=fast_forward(d,maxf = p1)
3 vs=as.numeric(sub("X","",as.character(fs$model)))
4 fdp(beta1,vs)
```

Listing B.4: Aplicación de Stepwise

```
1 qR=cv.glmnet(X1,Y1,alpha = 0)
2 LR=qR$lambda.min
3 Ridge <- glmnet(
4   X1          ,
5   Y1           ,
6   alpha = 0    ,
7   lambda      = LR,
8   standardize = TRUE
9 )
10 fdp(beta1,vs)
```

Listing B.5: Aplicacion de Regresion Ridge

```
1 qL=cv.glmnet(X1,Y1,alpha = 1)
2 plot(qL)
3 LL=qL$lambda.min
4 mlasso <- glmnet(
5   X1          ,
6   Y1           ,
7   alpha = 1    ,
8   lambda      = LL,
9   standardize = TRUE
10 )
11 fdp(beta1,which(mlasso$beta>0))
```

Listing B.6: Aplicación de Regresión Lasso

```
1 mknockoff1 = knockoff.filter(X1, Y1)
2 mknockoff1$selected
```

Listing B.7: Aplicación del metodo knockocff

Regresión lineal en dimensiona alta

```r
library(glmnet)
library(knockoff)
library(dplyr)
library(readxl)
library(data.table)
library(bigstep)
library(openxlsx)
library(ggplot2)
library(openxlsx)
library(readxl)
library(doMC)
library(multtest)
```

Listing B.8: Librerías

```r
set.seed(3445)

n2=100          # Número de observaciones
p2=300          # Número de variables
k2=30           # Número de variables con coeficientes distintos de cero.

# Generar las variables a partir de una distribución normal multivariada.

mu2 = rep(0,p2)
rho2 = 0.3
Sigma2 = toeplitz(rho2^(0:(p2-1)))
X2 = matrix(rnorm(n2*p2),n2) %*% chol(Sigma2)

# Generate the response from a linear model

nonzero2 = sample(p2, k2)
beta2 = 100 * (1:p2 %in% nonzero2)
Y2 = X2 %*% beta2 + rnorm(n2)
```

Listing B.9: Creación de la regresión lineal

```r
d2=prepare_data(Y2, X2)
fs2=fast_forward(d2,maxf = p2)
vs2=as.numeric(sub("X","",as.character(fs2$model)))
fdp(beta2,vs2)
```

Listing B.10: Stepwise

```r
qR=cv.glmnet(X2,Y2,alpha = 0)
LR=qR$lambda.min
Ridge2 <- glmnet(
  X2          ,
  Y2           ,
  alpha = 0    ,
  lambda      = LR,
  standardize = TRUE
)
```

Listing B.11: Regresion Ridge

```
qL2=cv.glmnet(X2,Y2,alpha = 1)
LL2=qL2$lambda.min
mlasso2 <- glmnet(
  X2          ,
  Y2          ,
  alpha = 1   ,
  lambda      = LL2,
  standardize = TRUE
)
fdp(beta2,which(mlasso2$beta>0))
```

Listing B.12: lasso

```
mknockoff2 = knockoff.filter(X2, Y2)
mknockoff2$selected
```

Listing B.13: Aplicación del método knockocff

Regresión lineal para pruebas de hipótesis múltiples

```
set.seed(4567)

n1=200          # number of observations
p1=100          # number of variables
k1=30            # number of variables with nonzero coefficients

# Generate the variables from a multivariate normal distribution

mu1 = rep(0,p1)
rho1 = 0.3
Sigma1 = toeplitz(rho1^(0:(p1-1)))
X1 = matrix(rnorm(n1*p1),n1) %*% chol(Sigma1)

# Generate the response from a linear model

nonzero1 = sample(p1, k1)
beta1 = 5 * (1:p1 %in% nonzero1)
Y1 = X1 %*% beta1 + rnorm(n1)
```

Listing B.14: creacion de la regresion linel

```
MCO=lm(Y1~X1)
summary(MCO)
pval = summary(MCO)$coefficients[-1,4]
Pval=as.numeric(pval)
OP=sort(Pval)
```

Listing B.15: Cálculos de loo p-valores

```
a=(1:p1)
c1=0.05/p1
windows()
```

```
plot(a,Pval,main="Bonferroni", xlab = "i", ylab = "p_i");
abline(h=c1,col="red")
abline(h=0.05,col="blue")
which(OP < c1)
which(Pval < c1)
Pval[c(4,10,15,19,20,24,31,36,37,40,42,44,58,61,65,66,67,68,70,
       75,76,79,80,85,87,93,94,97,99,100)]
```

Listing B.16: Bonforroni

```
c2=0.05/(p1-a+1)

#graficas

windows()
plot(a,OP,main="Bonferroni modificado (Homel)", xlab = "i", ylab = "p_(i)"
   );
abline(lsfit(a, c2),col="red")
abline(h=0.05,col="blue")
OP < c2
which(Pval < c2)
Pval[c(4,10,15,18,19,20,24,31,36,37,40,42,44,58,61,65,66,67,68,70,
       75,76,79,80,85,87,93,94,97,99,100)]
```

Listing B.17: Bonforroni modif

```
c3=a*(0.05/p1)
plot(a,c3,"l",col="red", xlab = "", ylab = "")
par(new=TRUE)
plot(a,OP,main="BH", xlab = "i", ylab = "p_(i)")

#graficas

windows()
plot(a,OP,main="Benjamini-Hochberg (BH)", xlab = "i", ylab = "p_(i)");
abline(lsfit(a, c3),col="red");abline(h=0.05,col="blue")
which(Pval < c3)
Pval[c(4,10,15,18,19,20,24,31,36,37,40,41,42,44,58,61,65,66,67,68,70,
       75,76,79,80,85,87,92,93,94,97,99,100)]
```

Listing B.18: Bengamine and Hochber

```
#fdp = function(selected) sum(beta[selected] == 0)/max(1,length(selected))
fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
   selected))

#vdp = function(selected) sum(beta[selected] != 0) / k
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

fdp(beta1,mknockoff1$selected)
vdp(beta1,mknockoff1$selected,k)

fdp(beta2,mknockoff2$selected)
vdp(beta2,mknockoff2$selected,k)
```

Listing B.19: Función para calcular el FDR y la potencia.

```r
library(MASS)

# Definir la funci?n de c?lculo del error tipo S

calc_error_S <- function(model, sigma, alpha = 0.05) {
  t_value <- qt(alpha/2, df = model$df.residual)
  beta <- model$coefficients[-1]
  se_beta <- sigma * sqrt(diag(solve(model$X)))
  ci_width <- 2 * t_value * se_beta
  width_ratio <- ci_width / abs(beta)
  max_width_ratio <- max(width_ratio)
  return(max_width_ratio)
}

# Cargar el conjunto de datos

data(Boston)

# Ajustar el modelo lineal
model <- lm(medv ~ ., data = Boston)

# Calcular el error tipo S
sigma <- summary(model)$sigma
error_S <- calc_error_S(model, sigma, alpha = 0.05)
print(paste0("Error tipo S = ", error_S))


library(meta)

# Cargar el conjunto de datos
data(Boston)

# Ajustar el modelo lineal
model <- lm(medv ~ ., data = Boston)

# Calcular el error tipo S
error_S <- sens.s(model)
print(paste0("Error tipo S = ", error_S))

sens_slope(model)
```

Listing B.20: Error tipo S

Simulación para dimensiona baja

```r
library(glmnet)
library(knockoff)
library(dplyr)
library(readxl)
```

```r
library(data.table)
library(bigstep)
library(openxlsx)
library(ggplot2)
library(openxlsx)
library(readxl)
```

Listing B.21: Librerías

```r
set.seed(4567)

n=3000
p=1500
NI=400
na=10
k=30
rho = 0.3
Sigma = toeplitz(rho^(0:(p-1)))

AFDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AFDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AFDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:na) {
  amplitudej = j*50
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = amplitudej * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    AFDRK[j,i]= fdp(betai,mknockoffi$selected)
    #Potencia
    AVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
```

```
    )
    AFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    AVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di,maxf = p)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    AFDRS[j,i]= fdp(betai,vsi)
    AVDRS[j,i]= vdp(betai,vsi,k)
  }
}

#FDR de las pruebas variando la amplitud

write.xlsx(AFDRK, "AFDRK.xlsx")
write.xlsx(AFDRL, "AFDRL.xlsx")
write.xlsx(AFDRS, "AFDRS.xlsx")

#FDR del metodo knockoff
AFDRK
meanAFDRK=numeric(na)
for(i in 1:na){
  meanAFDRK[i] = mean(AFDRK[i,])
}
meanAFDRK
plot(meanAFDRK, type = "l")

#FDR del metodo Lasso
AFDRL
meanAFDRL=numeric(na)
for(i in 1:na){
  meanAFDRL[i] = mean(AFDRL[i,])
}
meanAFDRL
plot(meanAFDRL, type = "l")

#FDR del metodo stepwise
AFDRS
meanAFDRS=numeric(na)
for(i in 1:na){
  meanAFDRS[i] = mean(AFDRS[i,])
}
meanAFDRS
plot(meanAFDRS, type = "l")

#todos juntos
meanFDR=c(meanAFDRK,meanAFDRL,meanAFDRS)
ampli=rep(1:na,3)*100
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))
FDRA=data.frame(meanFDR,ampli,FDR)
ggplot(data=FDRA, aes(x = ampli, y = meanFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point()+
  ggtitle("Media del FDR por metodo")+
```

```
  xlab("Amplitud") +
  ylab("Media del FDR")

#VDR de las pruebas variando la amplitud

#exportar a Exel

write.xlsx(AVDRK, "AVDRK.xlsx")
write.xlsx(AVDRL, "AVDRL.xlsx")
write.xlsx(AVDRS, "AVDRS.xlsx")

#VFDR del metodo knockoff
AVDRK
meanAVDRK=numeric(na)
for(i in 1:na){
  meanAVDRK[i] = mean(AVDRK[i,])
}
meanAVDRK
plot(meanAVDRK, type = "l")

#VDR del metodo Lasso
AVDRL
meanAVDRL=numeric(na)
for(i in 1:na){
  meanAVDRL[i] = mean(AVDRL[i,])
}
meanAVDRL
plot(meanAVDRL, type = "l")

#VDR del metodo stepwise
AVDRS
meanAVDRS=numeric(na)
for(i in 1:na){
  meanAVDRS[i] = mean(AVDRS[i,])
}
meanAVDRS
plot(meanAVDRS, type = "l")

#todos juntos
meanVDR=c(meanAVDRK,meanAVDRL,meanAVDRS)
ampli=rep(1:na,3)*100
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))

VDRA=data.frame(meanVDR,ampli,Potencia)

ggplot(data=FDRA, aes(x = ampli, y = meanFDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.22: Variando la amplitud

```r
#librerias
library(glmnet)
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)

# Variando las betas distintas de cero

set.seed(3456)

n=3000
p=1500
a=50
rho=0.3
Sigma = toeplitz(rho^(0:(p-1)))
NI=400
nk=10

KFDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KFDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KFDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:nk) {
  kj= 10*j
  for(i in 1:NI) {
  # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
  # Generate the response from a linear model
    nonzeroi = sample(p, kj)
    betai = a* (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    KFDRK[j,i]= fdp(betai,mknockoffi$selected)
    #Potencia
    KVDRK[j,i]= vdp(betai,mknockoffi$selected,kj)
    #lasss
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
```

```
      Xi          ,
      yi           ,
      lambda     = Li,
      standardize  = TRUE
    )

   KFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
   KVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),kj)
   #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    KFDRS[j,i]= fdp(betai,vsi)
    KVDRS[j,i]= vdp(betai,vsi,kj)
  }
}

#FDR de las pruebas variando las betas distintas de cero.
#exportar a Exel

write.xlsx(KFDRK, "KFDRK.xlsx")
write.xlsx(KFDRL, "KFDRL.xlsx")
write.xlsx(KFDRS, "KFDRS.xlsx")

#FDR del metodo knockoff

KFDRK=as.matrix(KFDRK)
meanKFDRK=numeric(nk)
for(i in 1:nk){
  meanKFDRK[i] = mean(KFDRK[i,])
}
meanKFDRK
plot(meanKFDRK, type = "l")

#FDR del metodo Lasso
KFDRL=as.matrix(KFDRL)
meanKFDRL=numeric(nk)
for(i in 1:nk){
  meanKFDRL[i] = mean(KFDRL[i,])
}
meanKFDRL
plot(meanKFDRL, type = "l")

#FDR del metodo stepwise
KFDRS=as.matrix(KFDRS)
meanKFDRS=numeric(nk)
for(i in 1:nk){
  meanKFDRS[i] = mean(KFDRS[i,])
}
meanKFDRS
plot(meanKFDRS, type = "l")

#todas las graficas juntas

```

```
meanKFDR=c(meanKFDRK,meanKFDRL,meanKFDRS)
bet=rep(1:nk,3)*10
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
FDRK=data.frame(meanKFDR,bet,FDR)
ggplot(data=FDRK, aes(x = bet, y = meanKFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
  ggtitle("Media del FDR por metodo")+
  xlab("Num. de betas distintas de cero") +
  ylab("Media del FDR")

#VDR de las pruebas variando los betas distintos de cero

#exportar a Exel
write.xlsx(KVDRK, "KVDRK.xlsx")
write.xlsx(KVDRL, "KVDRL.xlsx")
write.xlsx(KVDRS, "KVDRS.xlsx")

#VFDR del metodo knockoff
KVDRK
meanKVDRK=numeric(nk)
for(i in 1:nk){
  meanKVDRK[i] = mean(KVDRK[i,])
}
meanKVDRK
plot(meanKVDRK, type = "l")

#VDR del metodo Lasso
KVDRL
meanKVDRL=numeric(nk)
for(i in 1:nk){
  meanKVDRL[i] = mean(KVDRL[i,])
}
meanKVDRL
plot(meanKVDRL, type = "l")

#VDR del metodo stepwise
KVDRS
meanKVDRS=numeric(nk)
for(i in 1:nk){
  meanKVDRS[i] = mean(KVDRS[i,])
}
meanKVDRS
plot(meanKVDRS, type = "l")

#todos juntos

meanVDR=c(meanKVDRK,meanKVDRL,meanKVDRS)
beta0=rep(1:nk,3)*10
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
VDRK=data.frame(meanVDR,beta0,Potencia)
ggplot(data=VDRK, aes(x = beta0, y = meanVDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
```

```
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.23: Variando las betas distintas de cero

```

#librerias
library(glmnet)
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)

#Variando RHO

set.seed(5678)

n=3000
p=1500
a=50
k=30
NI=400
nr=1

RFDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RFDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RFDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:nr) {
  rhoj = (j/nr) - 0.05
  Sigma = toeplitz(rhoj^(0:(p-1)))
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = a * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
```

```
    RFDRK[j,i]= fdp(betai,mknockoffi$selected)
    RVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi          ,
      lambda      = Li,
      standardize = TRUE
    )
    RFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    RVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    RFDRS[j,i]= fdp(betai,vsi)
    RVDRS[j,i]= vdp(betai,vsi,k)
  }
}

#FDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(RFDRK, "RFDRK.xlsx")
write.xlsx(RFDRL, "RFDRL.xlsx")
write.xlsx(RFDRS, "RFDRS.xlsx")

#FDR del metodo knockoff
RFDRK = as.matrix(RFDRK)
meanRFDRK=numeric(nr)
for(i in 1:nr){
  meanRFDRK[i] = mean(RFDRK[i,])
}
meanRFDRK
plot(meanRFDRK, type = "l")

#FDR del metodo Lasso
RFDRL=as.matrix(RFDRL)
meanRFDRL=numeric(nr)
for(i in 1:nr){
  meanRFDRL[i] = mean(RFDRL[i,])
}
meanRFDRL
plot(meanRFDRL, type = "l")

#FDR del metodo stepwise
RFDRS=as.matrix(RFDRS)
meanRFDRS=numeric(nr)
for(i in 1:nr){
  meanRFDRS[i] = mean(RFDRS[i,])
}
meanRFDRS
```

```
plot(meanRFDRS, type = "l")

#todas las graficas juntas

meanRFDR=c(meanRFDRK,meanRFDRL,meanRFDRS)
R=rep(1:nr,3)/nr -0.05
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
FDRR=data.frame(meanRFDR,R,FDR)
ggplot(data=FDRR, aes(x =R , y = meanRFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
  ggtitle("Media del FDR por metodo")+
  xlab("RHO") +
  ylab("Media del FDR")

#leer los datos en R

RFDRK <- read_excel("RFDRK.xlsx")
RFDRL <- read_excel("RFDRL.xlsx")
RFDRS <- read_excel("RFDRS.xlsx")

#VDR de las pruebas variando la correlaci?n
#exportar a Exel

write.xlsx(RVDRK, "RVDRK.xlsx")
write.xlsx(RVDRL, "RVDRL.xlsx")
write.xlsx(RVDRS, "RVDRS.xlsx")

#VFDR del metodo knockoff
RVDRK
meanRVDRK=numeric(nr)
for(i in 1:nr){
  meanRVDRK[i] = mean(RVDRK[i,])
}
meanRVDRK
plot(meanRVDRK, type = "l")

#VDR del metodo Lasso
RVDRL
meanRVDRL=numeric(nr)
for(i in 1:nr){
  meanRVDRL[i] = mean(RVDRL[i,])
}
meanRVDRL
plot(meanRVDRL, type = "l")

#VDR del metodo stepwise
RVDRS
meanRVDRS=numeric(nr)
for(i in 1:nr){
  meanRVDRS[i] = mean(RVDRS[i,])
}
meanRVDRS
plot(meanRVDRS, type = "l")
```

```
155
156 #todos juntos
157 meanVDR=c(meanRVDRK,meanRVDRL,meanRVDRS)
158 rho=rep(1:nr,3)*(1/nr) - 0.05
159 Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
160 VDRR=data.frame(meanVDR,rho,Potencia)
161 ggplot(data=VDRR, aes(x = rho, y = meanVDR, group=Potencia, color=Potencia
    )) +
162   geom_line() +
163   geom_point()+
164   ggtitle("Medias de las Potencias")+
165   xlab("Amplitud") +
166   ylab("Medias de las potencias")
```

Listing B.24: Error tipo S

Simulación para dimensiona media

```
1  library(glmnet)
2  library(knockoff)
3  library(dplyr)
4  library(readxl)
5  library(data.table)
6  library(bigstep)
7  library(openxlsx)
8  library(ggplot2)
9  library(openxlsx)
10 library(readxl)
11 library(doMC)
12
13 #Variando la amplitud
14
15 set.seed(6789)
16
17 n=250
18 p=500
19 NI=3
20 na=3
21 k=30
22 rho = 0.3
23 Sigma = toeplitz(rho^(0:(p-1)))
24
25 AFDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
26 AFDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
27 AFDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)
28 AVDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
29 AVDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
30 AVDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)
31
32 fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
33 vdp = function(beta,selected,k) sum(beta[selected] != 0) / k
34
```

```r
for (j in 1:na) {
  amplitudej = j*20
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = amplitudej * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    AFDRK[j,i]= fdp(betai,mknockoffi$selected)
    #Potencia
    AVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
    )
    AFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    AVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di,maxf = p)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    AFDRS[j,i]= fdp(betai,vsi)
    AVDRS[j,i]= vdp(betai,vsi,k)
  }
}

#FDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(AFDRK, "AFDRK.xlsx")
write.xlsx(AFDRL, "AFDRL.xlsx")
write.xlsx(AFDRS, "AFDRS.xlsx")

#FDR del metodo knockoff
AFDRK
meanAFDRK=numeric(na)
for(i in 1:na){
  meanAFDRK[i] = mean(AFDRK[i,])
}
meanAFDRK
plot(meanAFDRK, type = "l")

#FDR del metodo Lasso
AFDRL
```

```
meanAFDRL=numeric(na)
for(i in 1:na){
  meanAFDRL[i] = mean(AFDRL[i,])
}
meanAFDRL
plot(meanAFDRL, type = "l")

#FDR del metodo stepwise
AFDRS
meanAFDRS=numeric(na)
for(i in 1:na){
  meanAFDRS[i] = mean(AFDRS[i,])
}
meanAFDRS
plot(meanAFDRS, type = "l")

#todos juntos
meanFDR=c(meanAFDRK,meanAFDRL,meanAFDRS)
ampli=rep(1:na,3)*100
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))
FDRA=data.frame(meanFDR,ampli,FDR)
ggplot(data=FDRA, aes(x = ampli, y = meanFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point()+
  ggtitle("Media del FDR por metodo")+
  xlab("Amplitud") +
  ylab("Media del FDR")

#VDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(AVDRK, "AVDRK.xlsx")
write.xlsx(AVDRL, "AVDRL.xlsx")
write.xlsx(AVDRS, "AVDRS.xlsx")

#VFDR del metodo knockoff
AVDRK
meanAVDRK=numeric(na)
for(i in 1:na){
  meanAVDRK[i] = mean(AVDRK[i,])
}
meanAVDRK
plot(meanAVDRK, type = "l")

#VDR del metodo Lasso
AVDRL
meanAVDRL=numeric(na)
for(i in 1:na){
  meanAVDRL[i] = mean(AVDRL[i,])
}
meanAVDRL
plot(meanAVDRL, type = "l")

#VDR del metodo stepwise
```

```
AVDRS
meanAVDRS=numeric(na)
for(i in 1:na){
  meanAVDRS[i] = mean(AVDRS[i,])
}
meanAVDRS
plot(meanAVDRS, type = "l")

#todos juntos

meanVDR=c(meanAVDRK,meanAVDRL,meanAVDRS)
ampli=rep(1:na,3)*100
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))
VDRA=data.frame(meanVDR,ampli,Potencia)
ggplot(data=FDRA, aes(x = ampli, y = meanFDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.25: Variando la amplitud

```
library(glmnet)
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)

#Variando las betas distintas de cero

set.seed(5678)

n=250
p=500
a=
rho=0.3
Sigma = toeplitz(rho^(0:(p-1)))
NI=400
nk=10

KFDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KFDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KFDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
KVDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
```

```
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:nk) {
  kj= 10*j
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, kj)
    betai = a* (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    KFDRK[j,i]= fdp(betai,mknockoffi$selected)
    #Potencia
    KVDRK[j,i]= vdp(betai,mknockoffi$selected,kj)
    #lasss
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
    )
   KFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
   KVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),kj)
   #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    KFDRS[j,i]= fdp(betai,vsi)
    KVDRS[j,i]= vdp(betai,vsi,kj)
  }
}

#FDR de las pruebas variando las betas distintas de cero.
#exportar a Exel

write.xlsx(KFDRK, "KFDRK.xlsx")
write.xlsx(KFDRL, "KFDRL.xlsx")
write.xlsx(KFDRS, "KFDRS.xlsx")

#FDR del metodo knockoff
KFDRK=as.matrix(KFDRK)
meanKFDRK=numeric(nk)
for(i in 1:nk){
  meanKFDRK[i] = mean(KFDRK[i,])
}
meanKFDRK
plot(meanKFDRK, type = "l")

```

```
#FDR del metodo Lasso
KFDRL=as.matrix(KFDRL)
meanKFDRL=numeric(nk)
for(i in 1:nk){
  meanKFDRL[i] = mean(KFDRL[i,])
}
meanKFDRL
plot(meanKFDRL, type = "l")

#FDR del metodo stepwise
KFDRS=as.matrix(KFDRS)
meanKFDRS=numeric(nk)
for(i in 1:nk){
  meanKFDRS[i] = mean(KFDRS[i,])
}
meanKFDRS
plot(meanKFDRS, type = "l")

#todas las graficas juntas

meanKFDR=c(meanKFDRK,meanKFDRL,meanKFDRS)
bet=rep(1:nk,3)*10
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
FDRK=data.frame(meanKFDR,bet,FDR)
ggplot(data=FDRK, aes(x = bet, y = meanKFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
  ggtitle("Media del FDR por metodo")+
  xlab("Num. de betas distintas de cero") +
  ylab("Media del FDR")

#leer en R

KFDRK <- read_excel("KFDRK.xlsx")
KFDRL <- read_excel("KFDRL.xlsx")
KFDRS <- read_excel("KFDRS.xlsx")

#VDR de las pruebas variando los betas distintos de cero
#exportar a Exel

write.xlsx(KVDRK, "KVDRK.xlsx")
write.xlsx(KVDRL, "KVDRL.xlsx")
write.xlsx(KVDRS, "KVDRS.xlsx")

#VFDR del metodo knockoff
KVDRK
meanKVDRK=numeric(nk)
for(i in 1:nk){
  meanKVDRK[i] = mean(KVDRK[i,])
}
meanKVDRK
plot(meanKVDRK, type = "l")

#VDR del metodo Lasso
```

```
KVDRL
meanKVDRL=numeric(nk)
for(i in 1:nk){
  meanKVDRL[i] = mean(KVDRL[i,])
}
meanKVDRL
plot(meanKVDRL, type = "l")

#VDR del metodo stepwise
KVDRS
meanKVDRS=numeric(nk)
for(i in 1:nk){
  meanKVDRS[i] = mean(KVDRS[i,])
}
meanKVDRS
plot(meanKVDRS, type = "l")

#todos juntos

meanVDR=c(meanKVDRK,meanKVDRL,meanKVDRS)
beta0=rep(1:nk,3)*10
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
VDRK=data.frame(meanVDR,beta0,Potencia)
ggplot(data=VDRK, aes(x = beta0, y = meanVDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.26: Variando las betas distintas de cero

```
library(glmnet)
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)

#Variando RHO

set.seed(6789)

n=250
p=500
a=
k=30
NI=400
nr=10

RFDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
```

```
RFDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RFDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k
for (j in 1:nr) {
  rhoj = (j/nr) - 0.05
  Sigma = toeplitz(rhoj^(0:(p-1)))
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = a * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    RFDRK[j,i]= fdp(betai,mknockoffi$selected)
    RVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
    )
    RFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    RVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    RFDRS[j,i]= fdp(betai,vsi)
    RVDRS[j,i]= vdp(betai,vsi,k)
  }
}

#FDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(RFDRK, "RFDRK.xlsx")
write.xlsx(RFDRL, "RFDRL.xlsx")
write.xlsx(RFDRS, "RFDRS.xlsx")

#FDR del metodo knockoff
RFDRK = as.matrix(RFDRK)
```

```
meanRFDRK=numeric(nr)
for(i in 1:nr){
  meanRFDRK[i] = mean(RFDRK[i,])
}
meanRFDRK
plot(meanRFDRK, type = "l")

#FDR del metodo Lasso
RFDRL=as.matrix(RFDRL)
menRFDRL=numeric(nr)
for(i in 1:nr){
  meanRFDRL[i] = mean(RFDRL[i,])
}
meanRFDRL
plot(meanRFDRL, type = "l")

#FDR del metodo stepwise
RFDRS=as.matrix(RFDRS)
meanRFDRS=numeric(nr)
for(i in 1:nr){
  meanRFDRS[i] = mean(RFDRS[i,])
}
meanRFDRS
plot(meanRFDRS, type = "l")

#todas las graficas juntas
meanRFDR=c(meanRFDRK,meanRFDRL,meanRFDRS)
R=rep(1:nr,3)/nr -0.05
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
FDRR=data.frame(meanRFDR,R,FDR)
ggplot(data=FDRR, aes(x =R , y = meanRFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
  ggtitle("Media del FDR por metodo")+
  xlab("RHO") +
  ylab("Media del FDR")

#leer los datos en R

RFDRK <- read_excel("RFDRK.xlsx")
RFDRL <- read_excel("RFDRL.xlsx")
RFDRS <- read_excel("RFDRS.xlsx")

#VDR de las pruebas variando la correlación
#exportar a Exel

write.xlsx(RVDRK, "RVDRK.xlsx")
write.xlsx(RVDRL, "RVDRL.xlsx")
write.xlsx(RVDRS, "RVDRS.xlsx")

#VFDR del metodo knockoff
RVDRK
meanRVDRK=numeric(nr)
for(i in 1:nr){
```

```
  meanRVDRK[i] = mean(RVDRK[i,])
}
meanRVDRK
plot(meanRVDRK, type = "l")

#VDR del metodo Lasso
RVDRL
meanRVDRL=numeric(nr)
for(i in 1:nr){
  meanRVDRL[i] = mean(RVDRL[i,])
}
meanRVDRL
plot(meanRVDRL, type = "l")

#VDR del metodo stepwise
RVDRS
meanRVDRS=numeric(nr)
for(i in 1:nr){
  meanRVDRS[i] = mean(RVDRS[i,])
}
meanRVDRS
plot(meanRVDRS, type = "l")

#todos juntos
meanVDR=c(meanRVDRK,meanRVDRL,meanRVDRS)
rho=rep(1:nr,3)*(1/nr) - 0.05
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
VDRR=data.frame(meanVDR,rho,Potencia)
ggplot(data=VDRR, aes(x = rho, y = meanVDR, group=Potencia, color=Potencia
    )) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.27: variando rho

Simulacion para dimencion alta

```
library(glmnet)
library(knockoff)
library(dplyr)
library(readxl)
library(data.table)
library(bigstep)
library(openxlsx)
library(ggplot2)
library(doMC)

#Variando la amplitud

set.seed(6789)
```

```r

n=250
p=1200
NI=3
na=3
k=30
rho = 0.3
Sigma = toeplitz(rho^(0:(p-1)))

AFDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AFDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AFDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRK=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRL=matrix(rep(0,p*NI),nrow=na,ncol = NI)
AVDRS=matrix(rep(0,p*NI),nrow=na,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:na) {
  amplitudej = j*20
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = amplitudej * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    AFDRK[j,i]= fdp(betai,mknockoffi$selected)
    #Potencia
    AVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
    )
    AFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    AVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di,maxf = p)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    AFDRS[j,i]= fdp(betai,vsi)
    AVDRS[j,i]= vdp(betai,vsi,k)
  }
```

```
}

#FDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(AFDRK, "AFDRK.xlsx")
write.xlsx(AFDRL, "AFDRL.xlsx")
write.xlsx(AFDRS, "AFDRS.xlsx")

#FDR del metodo knockoff
AFDRK
meanAFDRK=numeric(na)
for(i in 1:na){
  meanAFDRK[i] = mean(AFDRK[i,])
}
meanAFDRK
plot(meanAFDRK, type = "l")

#FDR del metodo Lasso
AFDRL
meanAFDRL=numeric(na)

for(i in 1:na){
  meanAFDRL[i] = mean(AFDRL[i,])
}
meanAFDRL
plot(meanAFDRL, type = "l")

#FDR del metodo stepwise
AFDRS
meanAFDRS=numeric(na)
for(i in 1:na){
  meanAFDRS[i] = mean(AFDRS[i,])
}
meanAFDRS
plot(meanAFDRS, type = "l")

#todos juntos

meanFDR=c(meanAFDRK,meanAFDRL,meanAFDRS)
ampli=rep(1:na,3)*100
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))
FDRA=data.frame(meanFDR,ampli,FDR)
ggplot(data=FDRA, aes(x = ampli, y = meanFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point()+
  ggtitle("Media del FDR por metodo")+
  xlab("Amplitud") +
  ylab("Media del FDR")

#VDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(AVDRK, "AVDRK.xlsx")
```

```
write.xlsx(AVDRL, "AVDRL.xlsx")
write.xlsx(AVDRS, "AVDRS.xlsx")

#VFDR del metodo knockoff
AVDRK
meanAVDRK=numeric(na)
for(i in 1:na){
  meanAVDRK[i] = mean(AVDRK[i,])
}
meanAVDRK
plot(meanAVDRK, type = "l")

#VDR del metodo Lasso
AVDRL
meanAVDRL=numeric(na)
for(i in 1:na){
  meanAVDRL[i] = mean(AVDRL[i,])
}
meanAVDRL
plot(meanAVDRL, type = "l")

#VDR del metodo stepwise
AVDRS
meanAVDRS=numeric(na)
for(i in 1:na){
  meanAVDRS[i] = mean(AVDRS[i,])
}
meanAVDRS
plot(meanAVDRS, type = "l")

#todos juntos

meanVDR=c(meanAVDRK,meanAVDRL,meanAVDRS)
ampli=rep(1:na,3)*100
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(na,na,na))
VDRA=data.frame(meanVDR,ampli,Potencia)
ggplot(data=FDRA, aes(x = ampli, y = meanVDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.28: Variando la amplitud

```
library(glmnet)
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)
```

```
9
10   #Variando las betas distintas de cero
11
12 set.seed(3456)
13
14 n=250
15 p=1500
16 a=
17 rho=
18 Sigma = toeplitz(rho^(0:(p-1)))
19 NI=400
20 nk=10
21
22 KFDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
23 KFDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
24 KFDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
25 KVDRK=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
26 KVDRL=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
27 KVDRS=matrix(rep(0,p*NI),nrow=nk,ncol = NI)
28
29 fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
       selected))
30 vdp = function(beta,selected,k) sum(beta[selected] != 0) / k
31
32 for (j in 1:nk) {
33   kj= 10*j
34   for(i in 1:NI) {
35     # Generate the variables from a multivariate normal distribution
36     mui = rep(0,p)
37     Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
38     # Generate the response from a linear model
39     nonzeroi = sample(p, kj)
40     betai = a* (1:p %in% nonzeroi)
41     yi = Xi %*% betai + rnorm(n)
42     #knockocff
43     mknockoffi = knockoff.filter(Xi, yi)
44     #FDR
45     KFDRK[j,i]= fdp(betai,mknockoffi$selected)
46     #Potencia
47     KVDRK[j,i]= vdp(betai,mknockoffi$selected,kj)
48     #lasss
49     qi=cv.glmnet(Xi,yi)
50     Li=qi$lambda.min
51     mlasso1i <- glmnet(
52       Xi          ,
53       yi           ,
54       lambda      = Li,
55       standardize = TRUE
56     )
57    KFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
58    KVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),kj)
59    #Stepwise
60     di = prepare_data(yi, Xi)
61     fsi=fast_forward(di)
```

```
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    KFDRS[j,i]= fdp(betai,vsi)
    KVDRS[j,i]= vdp(betai,vsi,kj)
  }
}

#FDR de las pruebas variando las betas distintas de cero.
#exportar a Exel

write.xlsx(KFDRK, "KFDRK.xlsx")
write.xlsx(KFDRL,"KFDRL.xlsx")
write.xlsx(KFDRS, "KFDRS.xlsx")

#FDR del metodo knockoff
KFDRK=as.matrix(KFDRK)
meanKFDRK=numeric(nk)
for(i in 1:nk){
  meanKFDRK[i] = mean(KFDRK[i,])
}
meanKFDRK
plot(meanKFDRK, type = "l")

#FDR del metodo Lasso
KFDRL=as.matrix(KFDRL)
meanKFDRL=numeric(nk)
for(i in 1:nk){
  meanKFDRL[i] = mean(KFDRL[i,])
}
meanKFDRL
plot(meanKFDRL, type = "l")

#FDR del metodo stepwise
KFDRS=as.matrix(KFDRS)
meanKFDRS=numeric(nk)
for(i in 1:nk){
  meanKFDRS[i] = mean(KFDRS[i,])
}
meanKFDRS
plot(meanKFDRS, type = "l")

#todas las graficas juntas
meanKFDR=c(meanKFDRK,meanKFDRL,meanKFDRS)
bet=rep(1:nk,3)*10
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
FDRK=data.frame(meanKFDR,bet,FDR)
ggplot(data=FDRK, aes(x = bet, y = meanKFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
  ggtitle("Media del FDR por metodo")+
  xlab("Num. de betas distintas de cero") +
  ylab("Media del FDR")

#leer en R

```

```
KFDRK <- read_excel("KFDRK.xlsx")
KFDRL <- read_excel("KFDRL.xlsx")
KFDRS <- read_excel("KFDRS.xlsx")

#VDR de las pruebas variando los betas distintos de cero
#exportar a Exel

write.xlsx(KVDRK, "KVDRK.xlsx")
write.xlsx(KVDRL, "KVDRL.xlsx")
write.xlsx(KVDRS, "KVDRS.xlsx")

#VFDR del metodo knockoff
KVDRK
meanKVDRK=numeric(nk)
for(i in 1:nk){
  meanKVDRK[i] = mean(KVDRK[i,])
}
meanKVDRK
plot(meanKVDRK, type = "l")

#VDR del metodo Lasso
KVDRL
meanKVDRL=numeric(nk)
for(i in 1:nk){
  meanKVDRL[i] = mean(KVDRL[i,])
}
meanKVDRL
plot(meanKVDRL, type = "l")

#VDR del metodo stepwise
KVDRS
meanKVDRS=numeric(nk)
for(i in 1:nk){
  meanKVDRS[i] = mean(KVDRS[i,])
}
meanKVDRS
plot(meanKVDRS, type = "l")

#todos juntos
meanVDR=c(meanKVDRK,meanKVDRL,meanKVDRS)
beta0=rep(1:nk,3)*10
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nk,nk,nk))
VDRK=data.frame(meanVDR,beta0,Potencia)
ggplot(data=VDRK, aes(x = beta0, y = meanVDR, group=Potencia, color=
    Potencia)) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.29: Variando las betas distintas de cero

```
library(glmnet)
```

```
library(knockoff)
library(dplyr)
library(data.table)
library(bigstep)
library(ggplot2)
library(openxlsx)
library(readxl)

#Variando RHO

set.seed(5678)

n=250
p=1500
a=
k=30
NI=400
nr=10

RFDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RFDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RFDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRK=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRL=matrix(rep(0,p*NI),nrow=nr,ncol = NI)
RVDRS=matrix(rep(0,p*NI),nrow=nr,ncol = NI)

fdp = function(beta,selected) sum(beta[selected] == 0)/max(1,length(
    selected))
vdp = function(beta,selected,k) sum(beta[selected] != 0) / k

for (j in 1:nr) {
  rhoj = (j/nr) - 0.05
  Sigma = toeplitz(rhoj^(0:(p-1)))
  for(i in 1:NI) {
    # Generate the variables from a multivariate normal distribution
    mui = rep(0,p)
    Xi = matrix(rnorm(n*p),n) %*% chol(Sigma)
    # Generate the response from a linear model
    nonzeroi = sample(p, k)
    betai = a * (1:p %in% nonzeroi)
    yi = Xi %*% betai + rnorm(n)
    #knockocff
    mknockoffi = knockoff.filter(Xi, yi)
    #FDR
    RFDRK[j,i]= fdp(betai,mknockoffi$selected)
    RVDRK[j,i]= vdp(betai,mknockoffi$selected,k)
    #lasso
    qi=cv.glmnet(Xi,yi)
    Li=qi$lambda.min
    mlasso1i <- glmnet(
      Xi          ,
      yi           ,
      lambda      = Li,
      standardize = TRUE
```

```
    )
    RFDRL[j,i]= fdp(betai,which(mlasso1i$beta>0))
    RVDRL[j,i]= vdp(betai,which(mlasso1i$beta>0),k)
    #Stepwise
    di = prepare_data(yi, Xi)
    fsi=fast_forward(di)
    vsi=as.numeric(sub("X","",as.character(fsi$model)))
    RFDRS[j,i]= fdp(betai,vsi)
    RVDRS[j,i]= vdp(betai,vsi,k)
  }
}

#FDR de las pruebas variando la amplitud
#exportar a Exel

write.xlsx(RFDRK, "RFDRK.xlsx")
write.xlsx(RFDRL, "RFDRL.xlsx")
write.xlsx(RFDRS, "RFDRS.xlsx")

#FDR del metodo knockoff
RFDRK = as.matrix(RFDRK)
meanRFDRK=numeric(nr)
for(i in 1:nr){
  meanRFDRK[i] = mean(RFDRK[i,])
}
meanRFDRK
plot(meanRFDRK, type = "l")

#FDR del metodo Lasso
RFDRL=as.matrix(RFDRL)
meanRFDRL=numeric(nr)
for(i in 1:nr){
  meanRFDRL[i] = mean(RFDRL[i,])
}
meanRFDRL
plot(meanRFDRL, type = "l")

#FDR del metodo stepwise
RFDRS=as.matrix(RFDRS)
meanRFDRS=numeric(nr)
for(i in 1:nr){
  meanRFDRS[i] = mean(RFDRS[i,])
}
meanRFDRS
plot(meanRFDRS, type = "l")

#todas las graficas juntas
meanRFDR=c(meanRFDRK,meanRFDRL,meanRFDRS)
R=rep(1:nr,3)/nr -0.05
FDR=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
FDRR=data.frame(meanRFDR,R,FDR)
ggplot(data=FDRR, aes(x =R , y = meanRFDR, group = FDR, color = FDR)) +
  geom_line() +
  geom_point() +
```

```
  ggtitle("Media del FDR por metodo")+
  xlab("RHO") +
  ylab("Media del FDR")

#leer los datos en R

RFDRK <- read_excel("RFDRK.xlsx")
RFDRL <- read_excel("RFDRL.xlsx")
RFDRS <- read_excel("RFDRS.xlsx")

#VDR de las pruebas variando la correlación
#exportar a Exel

write.xlsx(RVDRK, "RVDRK.xlsx")
write.xlsx(RVDRL, "RVDRL.xlsx")
write.xlsx(RVDRS, "RVDRS.xlsx")

#VFDR del metodo knockoff
RVDRK
meanRVDRK=numeric(nr)
for(i in 1:nr){
  meanRVDRK[i] = mean(RVDRK[i,])
}
meanRVDRK
plot(meanRVDRK, type = "l")

#VDR del metodo Lasso
RVDRL
meanRVDRL=numeric(nr)
for(i in 1:nr){
  meanRVDRL[i] = mean(RVDRL[i,])
}
meanRVDRL
plot(meanRVDRL, type = "l")

#VDR del metodo stepwise
RVDRS
meanRVDRS=numeric(nr)
for(i in 1:nr){
  meanRVDRS[i] = mean(RVDRS[i,])
}
meanRVDRS
plot(meanRVDRS, type = "l")

#todos juntos
meanVDR=c(meanRVDRK,meanRVDRL,meanRVDRS)
rho=rep(1:nr,3)*(1/nr) - 0.05
Potencia=rep(c("Knockoff","Lasso","Stepwise"),c(nr,nr,nr))
VDRR=data.frame(meanVDR,rho,Potencia)
ggplot(data=VDRR, aes(x = rho, y = meanVDR, group=Potencia, color=Potencia
    )) +
  geom_line() +
  geom_point()+
  ggtitle("Medias de las Potencias")+
```

```
  xlab("Amplitud") +
  ylab("Medias de las potencias")
```

Listing B.30: Variando RHO

Aplicacion a datos reales

```

library(glmnet)
library(knockoff)
library(bigstep)
library(doMC)

#Obtencion y preparacion de datos.
drug_class = 'NNRTI' # Possible drug types are 'PI', 'NRTI', and 'NNRTI'.

#Primero, descargamos los datos y los leemos en marcos de datos.
base_url = 'http://hivdb.stanford.edu/pages/published_analysis/genophenoPNAS2006'
gene_url = paste(base_url, 'DATA', paste0(drug_class, '_DATA.txt'), sep='/')
tsm_url = paste(base_url, 'MUTATIONLISTS', 'NP_TSM', drug_class, sep='/')
gene_df = read.delim(gene_url, na.string = c('NA', ''), stringsAsFactors = FALSE)
tsm_df = read.delim(tsm_url, header = FALSE, stringsAsFactors = FALSE)
names(tsm_df) = c('Position', 'Mutations')

# Returns rows for which every column matches the given regular expression.
grepl_rows <- function(pattern, df) {
  cell_matches = apply(df, c(1,2), function(x) grepl(pattern, x))
  apply(cell_matches, 1, all)
}
pos_start = which(names(gene_df) == 'P1')
pos_cols = seq.int(pos_start, ncol(gene_df))
valid_rows = grepl_rows('^(\\.|-|[A-Zid]+)$', gene_df[,pos_cols])
gene_df = gene_df[valid_rows,]

#Ahora construimos la matriz de dise?o X y la matriz de vectores de respuesta
#Y. Las caracter?sticas (columnas de X ) est?n dadas por pares de
#mutaci?n (posici?n). Defina  [X_ {i, j} = {1 si el i ?simo paciente
#tiene el  j  ?simo par mutaci?n (posici?n) y 0 en caso contrario} , Y_ {i, j} =
#{resistencia del paciente  i  a la droga j }.] Por ejemplo, en la
#muestra de f?rmacos de tipo PI, se observan tres mutaciones diferentes (A, C y D)
#en la posici?n 63 de la proteasa, por lo que tres columnas de X ( denominadas
#P63.A, P63 .C y P63.D) indican la presencia o ausencia de cada mutaci?n en
#esta posici?n.

```

```r
# Flatten (aplanar) a matrix to a vector with names from concatenating
# row/column names.
flatten_matrix <- function(M, sep='.') {
  x <- c(M)
  names(x) <- c(outer(rownames(M), colnames(M),
                      function(...) paste(..., sep=sep)))
  x
}

# Construct preliminary design matrix.
muts = c(LETTERS,'i', 'd')
X = outer(muts, as.matrix(gene_df[,pos_cols]), Vectorize(grepl))
X = aperm(X, c(2,3,1))
dimnames(X)[[3]] <- muts
X = t(apply(X, 1, flatten_matrix))
mode(X) <- 'numeric'

# Remove any mutation/position pairs that never appear in the data.
X = X[,colSums(X) != 0]

# Extract response matrix.
Y = gene_df[,4:(pos_start-1)]

hist(Y[,1], breaks='FD')
hist(log(Y[,1]), breaks='FD')

#Uso del filtro knockoff

knockoff_and_bhq <- function (X, y, q) {
  # Log-transform the drug resistance measurements.
  y = log(y)
  # Remove patients with missing measurements.
  missing = is.na(y)
  y = y[!missing]
  X = X[!missing,]
  # Remove predictors that appear less than 3 times.
  #Elimina los predictores que aparecen menos de 3 veces.
  X = X[,colSums(X) >= 3]
  # Remove duplicate predictors.
  X = X[,colSums(abs(cor(X)-1) < 1e-4) == 1]
  # Run the knockoff filter.
  knock.gen = function(x) create.fixed(x, method='equi')
  result = knockoff.filter(X, y, fdr=fdr, knockoffs=knock.gen, statistic=
     stat.glmnet_lambdasmax)
  knockoff_selected = names(result$selected)
  # Run BHq.
  p = ncol(X)
  lm.fit = lm(y ~ X - 1) # no intercept
  p.values = coef(summary(lm.fit))[,4]
  cutoff = max(c(0, which(sort(p.values) <= fdr * (1:p) / p)))
  bhq_selected = names(which(p.values <= fdr * cutoff / p))
  list(Knockoff = knockoff_selected, BHq = bhq_selected)
}

```

```
fdr = 0.20
results = lapply(Y, function(y) knockoff_and_bhq(X, y, fdr))

print(results[1])
print(results[2])

get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)

comparisons <- lapply(results, function(drug) {
  lapply(drug, function(selected) {
    positions = unique(get_position(selected)) # remove possible
       duplicates
    discoveries = length(positions)
    false_discoveries = length(setdiff(positions, tsm_df$Position))
    list(true_discoveries = discoveries - false_discoveries,
         false_discoveries = false_discoveries,
         fdp = false_discoveries / max(1, discoveries))
  })
})

print(comparisons[1])

for (drug in names(comparisons)) {
  plot_data = do.call(cbind, comparisons[[drug]])
  plot_data = plot_data[c('true_discoveries','false_discoveries'),]
  barplot(as.matrix(plot_data), main = paste('Resistance to', drug),
          col = c('navy','orange'), ylim = c(0,40))
}

#DLV

#Seleccionar la variable X e y.
y1 = log(Y[,1])
missing = is.na(y1)
y1 = y1[!missing]
X_NNRTI_DLV = X[!missing,]
X_NNRTI_DLV = X_NNRTI_DLV[,colSums(X_NNRTI_DLV) >= 3]
X_NNRTI_DLV = X_NNRTI_DLV[,colSums(abs(cor(X_NNRTI_DLV)-1) < 1e-4) == 1]
dim(X_NNRTI_DLV)
length(y1)

#knockoff
knock.gen = function(x) create.fixed(x, method='equi')
fdr=0.2
get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)
result1.1 = knockoff.filter(X_NNRTI_DLV, y1, fdr=fdr, knockoffs=knock.gen,
                            statistic=stat.glmnet_lambdasmax)
positions1.1=unique(get_position(names(result1.1$selected)))
discoveries1.1 = length(positions1.1)
false_discoveries1.1 = length(setdiff(positions1.1, tsm_df$Position))
true_discoveries1.1 = discoveries1.1 - false_discoveries1.1

```

```
#B&H
p1 = ncol(X_NNRTI_DLV)
lm.fit1 = lm(y1 ~ X_NNRTI_DLV - 1) # no intercept
p.values1 = coef(summary(lm.fit1))[,4]
cutoff1 = max(c(0, which(sort(p.values1) <= fdr * (1:p1) / p1)))
bhq_selected1 = names(which(p.values1 <= fdr * cutoff1 / p1))
positions2.1=unique(get_position(bhq_selected1))
discoveries2.1 = length(positions2.1)
false_discoveries2.1 = length(setdiff(positions2.1, tsm_df$Position))
true_discoveries2.1 = discoveries2.1 - false_discoveries2.1

#Lassso
q1=cv.glmnet(X_NNRTI_DLV,y1)
L1=q1$lambda.min
mlasso1 <- glmnet(
  X_NNRTI_DLV          ,
  y1           ,
  lambda      = L1,
  standardize = TRUE
)
which(mlasso1$beta>0)
positions3.1=unique(get_position(colnames(X_NNRTI_DLV)[which(mlasso1$beta
   >0)]))
discoveries3.1 = length(positions3.1)
false_discoveries3.1 = length(setdiff(positions3.1, tsm_df$Position))
true_discoveries3.1 = discoveries3.1 - false_discoveries3.1

#Stepwise
p1=dim(X_NNRTI_DLV)[1]
d1 = prepare_data(y1, X_NNRTI_DLV)
fs1=fast_forward(d1,maxf = p1)
fs1$model
positions4.1=unique(get_position(fs1$model))
discoveries4.1 = length(positions4.1)
false_discoveries4.1 = length(setdiff(positions4.1, tsm_df$Position))
true_discoveries4.1 = discoveries4.1 - false_discoveries4.1

#Graficas
knockoof1=c(true_discoveries1.1,false_discoveries1.1)
BHq1=c(true_discoveries2.1,false_discoveries2.1)
Lasso1=c(true_discoveries3.1,false_discoveries3.1)
Stepwise1=c(true_discoveries4.1,false_discoveries4.1)
tabla1=data.frame(knockoof1,BHq1,Lasso1,Stepwise1)

barplot(as.matrix(tabla1),
        main = "DLV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,60))

#EFV

#Variables
y2 = log(Y[,2])
missing2 = is.na(y2)
```

```
y2 = y2[!missing2]
X_NNRTI_EFV = X[!missing2,]
X_NNRTI_EFV = X_NNRTI_EFV[,colSums(X_NNRTI_EFV) >= 3]
X_NNRTI_EFV = X_NNRTI_EFV[,colSums(abs(cor(X_NNRTI_EFV)-1) < 1e-4) == 1]
dim(X_NNRTI_EFV)
length(y2)

#knockoof
result2 = knockoff.filter(X_NNRTI_EFV, y2, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.2=unique(get_position(names(result2$selected)))
discoveries1.2 = length(positions1.2)
false_discoveries1.2 = length(setdiff(positions1.2, tsm_df$Position))
true_discoveries1.2 = discoveries1.2 - false_discoveries1.2

#B&H
p2 = ncol(X_NNRTI_EFV)
lm.fit2 = lm(y2 ~ X_NNRTI_EFV - 1) # no intercept
p.values2 = coef(summary(lm.fit2))[,4]
cutoff2 = max(c(0, which(sort(p.values2) <= fdr * (1:p2) / p2)))
bhq_selected2 = names(which(p.values2 <= fdr * cutoff2 / p2))
positions2.2=unique(get_position(bhq_selected2))
discoveries2.2 = length(positions2.2)
false_discoveries2.2 = length(setdiff(positions2.2, tsm_df$Position))
true_discoveries2.2 = discoveries2.2 - false_discoveries2.2

#Lassso
q2=cv.glmnet(X_NNRTI_EFV,y2)
L2=q2$lambda.min
mlasso2 <- glmnet(
  X_NNRTI_EFV          ,
  y2           ,
  lambda      = L2,
  standardize = TRUE
)
which(mlasso2$beta>0)
positions3.2=unique(get_position(colnames(X_NNRTI_EFV)[which(mlasso2$beta
   >0)]))
discoveries3.2 = length(positions3.2)
false_discoveries3.2 = length(setdiff(positions3.2, tsm_df$Position))
true_discoveries3.2 = discoveries3.2 - false_discoveries3.2

#Stepwise
p2=dim(X_NNRTI_EFV)[1]
d2= prepare_data(y2, X_NNRTI_EFV)
fs2=fast_forward(d2,maxf = p2)
fs2$model
positions4.2=unique(get_position(fs2$model))
discoveries4.2 = length(positions4.2)
false_discoveries4.2 = length(setdiff(positions4.2, tsm_df$Position))
true_discoveries4.2 = discoveries4.2 - false_discoveries4.2

#Graficas
knockoof2=c(true_discoveries1.2,false_discoveries1.2)
```

```
BHq2=c(true_discoveries2.2,false_discoveries2.2)
Lasso2=c(true_discoveries3.2,false_discoveries3.2)
Stepwise2=c(true_discoveries4.2,false_discoveries4.2)
tabla2=data.frame(knockoof2,BHq2,Lasso2,Stepwise2)
barplot(as.matrix(tabla2),
        main = "EFV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,45))

#NVP

#Variables
y3 = log(Y[,3])
missing3 = is.na(y3)
y3 = y3[!missing3]
X_NNRTI_NVP = X[!missing3,]
X_NNRTI_NVP = X_NNRTI_NVP[,colSums(X_NNRTI_NVP) >= 3]
X_NNRTI_NVP = X_NNRTI_NVP[,colSums(abs(cor(X_NNRTI_NVP)-1) < 1e-4) == 1]
dim(X_NNRTI_NVP)
length(y3)

#knockoof
result3 = knockoff.filter(X_NNRTI_NVP, y3, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.3=unique(get_position(names(result3$selected)))
discoveries1.3 = length(positions1.3)
false_discoveries1.3 = length(setdiff(positions1.3, tsm_df$Position))
true_discoveries1.3 = discoveries1.3 - false_discoveries1.3

#B&H
p3 = ncol(X_NNRTI_NVP)
lm.fit3 = lm(y3 ~ X_NNRTI_NVP - 1) # no intercept
p.values3 = coef(summary(lm.fit3))[,4]
cutoff3 = max(c(0, which(sort(p.values3) <= fdr * (1:p3) / p3)))
bhq_selected3 = names(which(p.values3 <= fdr * cutoff3 / p3))
positions2.3=unique(get_position(bhq_selected3))
discoveries2.3 = length(positions2.3)
false_discoveries2.3 = length(setdiff(positions2.3, tsm_df$Position))
true_discoveries2.3 = discoveries2.3 - false_discoveries2.3

#Lassso
q3=cv.glmnet(X_NNRTI_NVP,y3)
L3=q3$lambda.min
mlasso3 <- glmnet(
  X_NNRTI_NVP      ,
  y3          ,
  lambda     = L3,
  standardize = TRUE
)
which(mlasso3$beta>0)
positions3.3=unique(get_position(colnames(X_NNRTI_NVP)[which(mlasso3$beta
    >0)]))
discoveries3.3 = length(positions3.3)
false_discoveries3.3 = length(setdiff(positions3.3, tsm_df$Position))
```

```
true_discoveries3.3 = discoveries3.3 - false_discoveries3.3

#Stepwise
p3=dim(X_NNRTI_NVP)[1]
d3= prepare_data(y3, X_NNRTI_NVP)
fs3=fast_forward(d3,maxf = p3)
fs3$model
positions4.3=unique(get_position(fs3$model))
discoveries4.3 = length(positions4.3)
false_discoveries4.3 = length(setdiff(positions4.3, tsm_df$Position))
true_discoveries4.3 = discoveries4.3 - false_discoveries4.3

#Graficas
knockoof3=c(true_discoveries1.3,false_discoveries1.3)
BHq3=c(true_discoveries2.3,false_discoveries2.3)
Lasso3=c(true_discoveries3.3,false_discoveries3.3)
Stepwise3=c(true_discoveries4.3,false_discoveries4.3)
tabla3=data.frame(knockoof3,BHq3,Lasso3,Stepwise3)
barplot(as.matrix(tabla3),
        main = "IDV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,50))
```

Listing B.31: NNRTI

```

library(glmnet)
library(knockoff)
library(bigstep)
library(doMC)

#Obtencion y preparacion de datos.
drug_class = 'NRTI' # Possible drug types are 'PI', 'NRTI', and 'NNRTI'.

#Primero, descargamos los datos y los leemos en marcos de datos.
base_url = 'http://hivdb.stanford.edu/pages/published_analysis/
    genophenoPNAS2006'
gene_url = paste(base_url, 'DATA', paste0(drug_class, '_DATA.txt'), sep='/
    ')
tsm_url = paste(base_url, 'MUTATIONLISTS', 'NP_TSM', drug_class, sep='/')
gene_df = read.delim(gene_url, na.string = c('NA', ''), stringsAsFactors =
     FALSE)
tsm_df = read.delim(tsm_url, header = FALSE, stringsAsFactors = FALSE)
names(tsm_df) = c('Position', 'Mutations')

# Returns rows for which every column matches the given regular expression
    .
grepl_rows <- function(pattern, df) {
  cell_matches = apply(df, c(1,2), function(x) grepl(pattern, x))
  apply(cell_matches, 1, all)
}
pos_start = which(names(gene_df) == 'P1')
pos_cols = seq.int(pos_start, ncol(gene_df))
valid_rows = grepl_rows('^(\\.|-|[A-Zid]+)$', gene_df[,pos_cols])
```

```
gene_df = gene_df[valid_rows,]

#Ahora construimos la matriz de dise?o X y la matriz de vectores de
    respuesta
#Y. Las caracter?sticas (columnas de X ) est?n dadas por pares de
#mutaci?n (posici?n). Defina  [X_ {i, j} = {1 si el i ?simo paciente
#tiene el  j ?simo par mutaci?n (posici?n) y 0 en caso contrario} , Y_ {i
    , j} =
#{resistencia del paciente  i  a la droga j }.] Por ejemplo, en la
#muestra de f?rmacos de tipo PI, se observan tres mutaciones diferentes (A
    , C y D)
#en la posici?n 63 de la proteasa, por lo que tres columnas de X (
    denominadas
#P63.A, P63 .C y P63.D) indican la presencia o ausencia de cada mutaci?n
    en
#esta posici?n.

# Flatten (aplanar) a matrix to a vector with names from concatenating
# row/column names.
flatten_matrix <- function(M, sep='.') {
  x <- c(M)
  names(x) <- c(outer(rownames(M), colnames(M),
                      function(...) paste(..., sep=sep)))
  x
}

# Construct preliminary design matrix.
muts = c(LETTERS, 'i', 'd')
X = outer(muts, as.matrix(gene_df[,pos_cols]), Vectorize(grepl))
X = aperm(X, c(2,3,1))
dimnames(X)[[3]] <- muts
X = t(apply(X, 1, flatten_matrix))
mode(X) <- 'numeric'

# Remove any mutation/position pairs that never appear in the data.
X = X[,colSums(X) != 0]

# Extract response matrix.
Y = gene_df[,4:(pos_start-1)]

hist(Y[,1], breaks='FD')
hist(log(Y[,1]), breaks='FD')

#Uso del filtro knockoff
knockoff_and_bhq <- function (X, y, q) {
  # Log-transform the drug resistance measurements.
  y = log(y)
  # Remove patients with missing measurements.
  missing = is.na(y)
  y = y[!missing]
  X = X[!missing,]
  # Remove predictors that appear less than 3 times.
  #Elimina los predictores que aparecen menos de 3 veces.
  X = X[,colSums(X) >= 3]
```

```
  # Remove duplicate predictors.
  X = X[,colSums(abs(cor(X)-1) < 1e-4) == 1]
  # Run the knockoff filter.
  knock.gen = function(x) create.fixed(x, method='equi')
  result = knockoff.filter(X, y, fdr=fdr, knockoffs=knock.gen, statistic=
     stat.glmnet_lambdasmax)
  knockoff_selected = names(result$selected)

  # Run BHq.
  p = ncol(X)
  lm.fit = lm(y ~ X - 1) # no intercept
  p.values = coef(summary(lm.fit))[,4]
  cutoff = max(c(0, which(sort(p.values) <= fdr * (1:p) / p)))
  bhq_selected = names(which(p.values <= fdr * cutoff / p))

  list(Knockoff = knockoff_selected, BHq = bhq_selected)
}

fdr = 0.20
results = lapply(Y, function(y) knockoff_and_bhq(X, y, fdr))
dim(X)
print(results[1])
print(results[2])
get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)

comparisons <- lapply(results, function(drug) {
  lapply(drug, function(selected) {
    positions = unique(get_position(selected)) # remove possible
       duplicates
    discoveries = length(positions)
    false_discoveries = length(setdiff(positions, tsm_df$Position))
    list(true_discoveries = discoveries - false_discoveries,
         false_discoveries = false_discoveries,
         fdp = false_discoveries / max(1, discoveries))
  })
})
print(comparisons[1])
for (drug in names(comparisons)) {
  plot_data = do.call(cbind, comparisons[[drug]])
  plot_data = plot_data[c('true_discoveries','false_discoveries'),]
  barplot(as.matrix(plot_data), main = paste('Resistance to', drug),
          col = c('navy','orange'), ylim = c(0,40))
}

#DLV

#Seleccionar la variable X e y.
y1 = log(Y[,1])
missing = is.na(y1)
y1 = y1[!missing]
X_NRTI_XTC = X[!missing,]
X_NRTI_XTC = X_NRTI_XTC[,colSums(X_NRTI_XTC) >= 3]
X_NRTI_XTC = X_NRTI_XTC[,colSums(abs(cor(X_NRTI_XTC)-1) < 1e-4) == 1]
```

```
dim(X_NRTI_XTC)
length(y1)

#knockoff
fdr=0.2
get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)
result1.1 = knockoff.filter(X_NRTI_XTC, y1, fdr=fdr,knockoffs=create.fixed
    ,
                            statistic=stat.glmnet_lambdasmax)
#u=knockoff.filter(X_NRTI_XTC, y1, fdr=fdr)
#u$selected
positions1.1=unique(get_position(names(result1.1$selected)))
discoveries1.1 = length(positions1.1)
false_discoveries1.1 = length(setdiff(positions1.1, tsm_df$Position))
true_discoveries1.1 = discoveries1.1 - false_discoveries1.1

#B&H
p1 = ncol(X_NRTI_XTC)
lm.fit1 = lm(y1 ~ X_NRTI_XTC - 1) # no intercept
p.values1 = coef(summary(lm.fit1))[,4]
cutoff1 = max(c(0, which(sort(p.values1) <= fdr * (1:p1) / p1)))
bhq_selected1 = names(which(p.values1 <= fdr * cutoff1 / p1))
positions2.1=unique(get_position(bhq_selected1))
discoveries2.1 = length(positions2.1)
false_discoveries2.1 = length(setdiff(positions2.1, tsm_df$Position))
true_discoveries2.1 = discoveries2.1 - false_discoveries2.1

#Lassso
q1=cv.glmnet(X_NRTI_XTC,y1)
L1=q1$lambda.min
mlasso1 <- glmnet(
  X_NRTI_XTC       ,
  y1           ,
  lambda     = L1,
  standardize = TRUE
)
which(mlasso1$beta>0)
positions3.1=unique(get_position(colnames(X_NRTI_XTC)[which(mlasso1$beta
    >0)]))
discoveries3.1 = length(positions3.1)
false_discoveries3.1 = length(setdiff(positions3.1, tsm_df$Position))
true_discoveries3.1 = discoveries3.1 - false_discoveries3.1

#Stepwise
p1=dim(X_NRTI_XTC)[1]
d1 = prepare_data(y1, X_NRTI_XTC)
fs1=fast_forward(d1,maxf = p1)
fs1$model
positions4.1=unique(get_position(fs1$model))
discoveries4.1 = length(positions4.1)
false_discoveries4.1 = length(setdiff(positions4.1, tsm_df$Position))
true_discoveries4.1 = discoveries4.1 - false_discoveries4.1

```

```r
#Graficas
knockoof1=c(true_discoveries1.1,false_discoveries1.1)
BHq1=c(true_discoveries2.1,false_discoveries2.1)
Lasso1=c(true_discoveries3.1,false_discoveries3.1)
Stepwise1=c(true_discoveries4.1,false_discoveries4.1)
tabla1=data.frame(knockoof1,BHq1,Lasso1,Stepwise1)
barplot(as.matrix(tabla1),
        main = "X3TC",
        xlab ="Metodo",
        col = c('navy','orange'), ylim = c(0,40))

#ABC

#Variables
y2 = log(Y[,2])
missing2 = is.na(y2)
y2 = y2[!missing2]
X_NRTI_ABC = X[!missing2,]
X_NRTI_ABC = X_NRTI_ABC[,colSums(X_NRTI_ABC) >= 3]
X_NRTI_ABC = X_NRTI_ABC[,colSums(abs(cor(X_NRTI_ABC)-1) < 1e-4) == 1]
dim(X_NRTI_ABC)
length(y2)

#knockoof
result2 = knockoff.filter(X_NRTI_ABC, y2, fdr=fdr, knockoffs= create.fixed
    ,
                          statistic=stat.glmnet_lambdasmax)
positions1.2=unique(get_position(names(result2$selected)))
discoveries1.2 = length(positions1.2)
false_discoveries1.2 = length(setdiff(positions1.2, tsm_df$Position))
true_discoveries1.2 = discoveries1.2 - false_discoveries1.2

#B&H

p2 = ncol(X_NRTI_ABC)
lm.fit2 = lm(y2 ~ X_NRTI_ABC - 1) # no intercept
p.values2 = coef(summary(lm.fit2))[,4]
cutoff2 = max(c(0, which(sort(p.values2) <= fdr * (1:p2) / p2)))
bhq_selected2 = names(which(p.values2 <= fdr * cutoff2 / p2))
positions2.2=unique(get_position(bhq_selected2))
discoveries2.2 = length(positions2.2)
false_discoveries2.2 = length(setdiff(positions2.2, tsm_df$Position))
true_discoveries2.2 = discoveries2.2 - false_discoveries2.2

#Lassso
q2=cv.glmnet(X_NRTI_ABC,y2)
L2=q2$lambda.min
mlasso2 <- glmnet(
  X_NRTI_ABC          ,
  y2           ,
  lambda      = L2,
  standardize = TRUE
)
which(mlasso2$beta>0)
```

```
positions3.2=unique(get_position(colnames(X_NRTI_ABC)[which(mlasso2$beta
    >0)]))
discoveries3.2 = length(positions3.2)
false_discoveries3.2 = length(setdiff(positions3.2, tsm_df$Position))
true_discoveries3.2 = discoveries3.2 - false_discoveries3.2

#Stepwise
p2=dim(X_NRTI_ABC)[1]
d2= prepare_data(y2, X_NRTI_ABC)
fs2=fast_forward(d2,maxf = p2)
fs2$model
positions4.2=unique(get_position(fs2$model))
discoveries4.2 = length(positions4.2)
false_discoveries4.2 = length(setdiff(positions4.2, tsm_df$Position))
true_discoveries4.2 = discoveries4.2 - false_discoveries4.2

#Graficas
knockoof2=c(true_discoveries1.2,false_discoveries1.2)
BHq2=c(true_discoveries2.2,false_discoveries2.2)
Lasso2=c(true_discoveries3.2,false_discoveries3.2)
Stepwise2=c(true_discoveries4.2,false_discoveries4.2)
tabla2=data.frame(knockoof2,BHq2,Lasso2,Stepwise2)
barplot(as.matrix(tabla2),
        main = "ABC",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,40))

#AZT

#Variables
y3 = log(Y[,3])
missing3 = is.na(y3)
y3 = y3[!missing3]
X_NRTI_AZT = X[!missing3,]
X_NRTI_AZT = X_NRTI_AZT[,colSums(X_NRTI_AZT) >= 3]
X_NRTI_AZT = X_NRTI_AZT[,colSums(abs(cor(X_NRTI_AZT)-1) < 1e-4) == 1]
dim(X_NRTI_AZT)
length(y3)

#knockoof
result3 = knockoff.filter(X_NRTI_AZT, y3, fdr=fdr, knockoffs=create.fixed,
                          statistic=stat.glmnet_lambdasmax)
positions1.3=unique(get_position(names(result3$selected)))
discoveries1.3 = length(positions1.3)
false_discoveries1.3 = length(setdiff(positions1.3, tsm_df$Position))
true_discoveries1.3 = discoveries1.3 - false_discoveries1.3

#B&H
p3 = ncol(X_NRTI_AZT)
lm.fit3 = lm(y3 ~ X_NRTI_AZT - 1) # no intercept
p.values3 = coef(summary(lm.fit3))[,4]
cutoff3 = max(c(0, which(sort(p.values3) <= fdr * (1:p3) / p3)))
bhq_selected3 = names(which(p.values3 <= fdr * cutoff3 / p3))
positions2.3=unique(get_position(bhq_selected3))
```

```
discoveries2.3 = length(positions2.3)
false_discoveries2.3 = length(setdiff(positions2.3, tsm_df$Position))
true_discoveries2.3 = discoveries2.3 - false_discoveries2.3

#Lassso
q3=cv.glmnet(X_NRTI_AZT,y3)
L3=q3$lambda.min
mlasso3 <- glmnet(
  X_NRTI_AZT     ,
  y3           ,
  lambda      = L3,
  standardize = TRUE
)
which(mlasso3$beta>0)
positions3.3=unique(get_position(colnames(X_NRTI_AZT)[which(mlasso3$beta
    >0)]))
discoveries3.3 = length(positions3.3)
false_discoveries3.3 = length(setdiff(positions3.3, tsm_df$Position))
true_discoveries3.3 = discoveries3.3 - false_discoveries3.3

#Stepwise
p3=dim(X_NRTI_AZT)[1]
d3= prepare_data(y3, X_NRTI_AZT)
fs3=fast_forward(d3,maxf = p3)
fs3$model
positions4.3=unique(get_position(fs3$model))
discoveries4.3 = length(positions4.3)
false_discoveries4.3 = length(setdiff(positions4.3, tsm_df$Position))
true_discoveries4.3 = discoveries4.3 - false_discoveries4.3

#Graficas
knockoof3=c(true_discoveries1.3,false_discoveries1.3)
BHq3=c(true_discoveries2.3,false_discoveries2.3)
Lasso3=c(true_discoveries3.3,false_discoveries3.3)
Stepwise3=c(true_discoveries4.3,false_discoveries4.3)
tabla3=data.frame(knockoof3,BHq3,Lasso3,Stepwise3)
barplot(as.matrix(tabla3),
        main = "AZT",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,55))

#D4T

#Seleccion de variables
y4 = log(Y[,4])
missing4 = is.na(y4)
y4 = y4[!missing4]
X_NRTI_DT = X[!missing4,]
X_NRTI_DT = X_NRTI_DT[,colSums(X_NRTI_DT) >= 3]
X_NRTI_DT = X_NRTI_DT[,colSums(abs(cor(X_NRTI_DT)-1) < 1e-4) == 1]
dim(X_NRTI_DT)
length(y4)

#knockoof
```

```
result4 = knockoff.filter(X_NRTI_DT, y4, fdr=fdr, knockoffs=create.fixed,
                          statistic=stat.glmnet_lambdasmax)
positions1.4=unique(get_position(names(result4$selected)))
discoveries1.4 = length(positions1.4)
false_discoveries1.4 = length(setdiff(positions1.4, tsm_df$Position))
true_discoveries1.4 = discoveries1.4 - false_discoveries1.4

#B&H
p4 = ncol(X_NRTI_DT)
lm.fit4 = lm(y4 ~ X_NRTI_DT - 1) # no intercept
p.values4 = coef(summary(lm.fit4))[,4]
cutoff4 = max(c(0, which(sort(p.values4) <= fdr * (1:p4) / p4)))
bhq_selected4 = names(which(p.values4 <= fdr * cutoff4 / p4))
positions2.4=unique(get_position(bhq_selected4))
discoveries2.4 = length(positions2.4)
false_discoveries2.4 = length(setdiff(positions2.4, tsm_df$Position))
true_discoveries2.4 = discoveries2.4 - false_discoveries2.4

#Lassso
q4=cv.glmnet(X_NRTI_DT,y4)
L4=q4$lambda.min
mlasso4 <- glmnet(
  X_NRTI_DT       ,
  y4          ,
  lambda     = L4,
  standardize = TRUE
)
which(mlasso4$beta>0)
positions3.4=unique(get_position(colnames(X_NRTI_DT)[which(mlasso4$beta>0)
   ]))
discoveries3.4 = length(positions3.4)
false_discoveries3.4 = length(setdiff(positions3.4, tsm_df$Position))
true_discoveries3.4 = discoveries3.4 - false_discoveries3.4

#Stepwise
p4=dim(X_NRTI_DT)[1]
d4= prepare_data(y4, X_NRTI_DT)
fs4=fast_forward(d4,maxf = p4)
fs4$model
positions4.4=unique(get_position(fs4$model))
discoveries4.4 = length(positions4.4)
false_discoveries4.4 = length(setdiff(positions4.4, tsm_df$Position))
true_discoveries4.4 = discoveries4.4 - false_discoveries4.4

#Graficas
knockoof4=c(true_discoveries1.4,false_discoveries1.4)
BHq4=c(true_discoveries2.4,false_discoveries2.4)
Lasso4=c(true_discoveries3.4,false_discoveries3.4)
Stepwise4=c(true_discoveries4.4,false_discoveries4.4)
tabla4=data.frame(knockoof4,BHq4,Lasso4,Stepwise4)
barplot(as.matrix(tabla4),
        main = "D4T",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,45))
```

```
391
392 #DDI
393
394 #Seleccion de variables
395 y5 = log(Y[,5])
396 missing5 = is.na(y5)
397 y5 = y5[!missing5]
398 X_NRTI_DDI = X[!missing5,]
399 X_NRTI_DDI = X_NRTI_DDI[,colSums(X_NRTI_DDI) >= 3]
400 X_NRTI_DDI = X_NRTI_DDI[,colSums(abs(cor(X_NRTI_DDI)-1) < 1e-4) == 1]
401 dim(X_NRTI_DDI)
402 length(y5)
403
404 #knockoof
405 result5 = knockoff.filter(X_NRTI_DDI, y5, fdr=fdr, knockoffs=create.fixed,
406                           statistic=stat.glmnet_lambdasmax)
407 positions1.5=unique(get_position(names(result5$selected)))
408 discoveries1.5 = length(positions1.5)
409 false_discoveries1.5 = length(setdiff(positions1.5, tsm_df$Position))
410 true_discoveries1.5 = discoveries1.5 - false_discoveries1.5
411
412 #B&H
413 p5 = ncol(X_NRTI_DDI)
414 lm.fit5 = lm(y5 ~ X_NRTI_DDI - 1) # no intercept
415 p.values5 = coef(summary(lm.fit5))[,4]
416 cutoff5 = max(c(0, which(sort(p.values5) <= fdr * (1:p5) / p5)))
417 bhq_selected5 = names(which(p.values5 <= fdr * cutoff5 / p5))
418 positions2.5=unique(get_position(bhq_selected5))
419 discoveries2.5 = length(positions2.5)
420 false_discoveries2.5 = length(setdiff(positions2.5, tsm_df$Position))
421 true_discoveries2.5 = discoveries2.5 - false_discoveries2.5
422
423 #Lassso
424 q5=cv.glmnet(X_NRTI_DDI,y5)
425 L5=q5$lambda.min
426 mlasso5 <- glmnet(
427   X_NRTI_DDI          ,
428   y5           ,
429   lambda      = L5,
430   standardize = TRUE
431 )
432 which(mlasso5$beta>0)
433 positions3.5=unique(get_position(colnames(X_NRTI_DDI)[which(mlasso5$beta
      >0)]))
434 discoveries3.5 = length(positions3.5)
435 false_discoveries3.5 = length(setdiff(positions3.5, tsm_df$Position))
436 true_discoveries3.5 = discoveries3.5 - false_discoveries3.5
437
438 #Stepwise
439 p5=dim(X_NRTI_DDI)[1]
440 d5= prepare_data(y5, X_NRTI_DDI)
441 fs5=fast_forward(d5,maxf = p5)
442 fs5$model
443 positions4.5=unique(get_position(fs5$model))
```

```
discoveries4.5 = length(positions4.5)
false_discoveries4.5 = length(setdiff(positions4.5, tsm_df$Position))
true_discoveries4.5 = discoveries4.5 - false_discoveries4.5

#Graficas
knockoof5=c(true_discoveries1.5,false_discoveries1.5)
BHq5=c(true_discoveries2.5,false_discoveries2.5)
Lasso5=c(true_discoveries3.5,false_discoveries3.5)
Stepwise5=c(true_discoveries4.5,false_discoveries4.5)
tabla5=data.frame(knockoof5,BHq5,Lasso5,Stepwise5)
barplot(as.matrix(tabla5),
        main = "DDI",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,40))

#TDF

#Seleccion de variables
y6 = log(Y[,6])
missing6 = is.na(y6)
y6 = y6[!missing6]
X_NRTI_TDF = X[!missing6,]
X_NRTI_TDF = X_NRTI_TDF[,colSums(X_NRTI_TDF) >= 3]
X_NRTI_TDF = X_NRTI_TDF[,colSums(abs(cor(X_NRTI_TDF)-1) < 1e-4) == 1]
dim(X_NRTI_TDF)
length(y6)

#knockoof

result6 = knockoff.filter(X_NRTI_TDF, y6, fdr=fdr, knockoffs=create.fixed,
                          statistic=stat.glmnet_lambdasmax)
positions1.6=unique(get_position(names(result6$selected)))
discoveries1.6 = length(positions1.6)
false_discoveries1.6 = length(setdiff(positions1.6, tsm_df$Position))
true_discoveries1.6 = discoveries1.6 - false_discoveries1.6

#B&H
p6 = ncol(X_NRTI_TDF)
lm.fit6 = lm(y6 ~X_NRTI_TDF - 1) # no intercept
p.values6 = coef(summary(lm.fit6))[,4]
cutoff6 = max(c(0, which(sort(p.values6) <= fdr * (1:p6) / p6)))
bhq_selected6 = names(which(p.values6 <= fdr * cutoff6 / p6))
positions2.6=unique(get_position(bhq_selected6))
discoveries2.6 = length(positions2.6)
false_discoveries2.6 = length(setdiff(positions2.6, tsm_df$Position))
true_discoveries2.6 = discoveries2.6 - false_discoveries2.6

#Lassso
q6=cv.glmnet(X_NRTI_TDF,y6)
L6=q6$lambda.min
mlasso6 <- glmnet(
  X_NRTI_TDF    ,
  y6          ,
  lambda      = L6,
```

```
  standardize = TRUE
)
which(mlasso6$beta>0)
positions3.6=unique(get_position(colnames(X_NRTI_TDF)[which(mlasso6$beta
    >0)]))
discoveries3.6 = length(positions3.6)
false_discoveries3.6 = length(setdiff(positions3.6, tsm_df$Position))
true_discoveries3.6 = discoveries3.6 - false_discoveries3.6

#Stepwise
p6=dim(X_NRTI_TDF)[1]
d6= prepare_data(y6, X_NRTI_TDF)
fs6=fast_forward(d6,maxf = p6)
fs6$model
positions4.6=unique(get_position(fs6$model))
discoveries4.6 = length(positions4.6)
false_discoveries4.6 = length(setdiff(positions4.6, tsm_df$Position))
true_discoveries4.6 = discoveries4.6 - false_discoveries4.6

#Graficas
knockoof6=c(true_discoveries1.6,false_discoveries1.6)
BHq6=c(true_discoveries2.6,false_discoveries2.6)
Lasso6=c(true_discoveries3.6,false_discoveries3.6)
Stepwise6=c(true_discoveries4.6,false_discoveries4.6)
tabla6=data.frame(knockoof6,BHq6,Lasso6,Stepwise6)
barplot(as.matrix(tabla6),
        main = "TDF",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,45))
```

Listing B.32: NRTI

```

library(glmnet)
library(knockoff)
library(bigstep)
library(doMC)

#Obtencion y preparacion de datos.
drug_class = 'PI' # Possible drug types are 'PI', 'NRTI', and 'NNRTI'.

#Primero, descargamos los datos y los leemos en marcos de datos.
base_url = 'http://hivdb.stanford.edu/pages/published_analysis/
    genophenoPNAS2006'
gene_url = paste(base_url, 'DATA', paste0(drug_class, '_DATA.txt'), sep='/
    ')
tsm_url = paste(base_url, 'MUTATIONLISTS', 'NP_TSM', drug_class, sep='/')
gene_df = read.delim(gene_url, na.string = c('NA', ''), stringsAsFactors =
     FALSE)
tsm_df = read.delim(tsm_url, header = FALSE, stringsAsFactors = FALSE)
names(tsm_df) = c('Position', 'Mutations')

# Returns rows for which every column matches the given regular expression
    .
```

```
grepl_rows <- function(pattern, df) {
  cell_matches = apply(df, c(1,2), function(x) grepl(pattern, x))
  apply(cell_matches, 1, all)
}
pos_start = which(names(gene_df) == 'P1')
pos_cols = seq.int(pos_start, ncol(gene_df))
valid_rows = grepl_rows('^(\\.|-|[A-Zid]+)$', gene_df[,pos_cols])
gene_df = gene_df[valid_rows,]

#Ahora construimos la matriz de dise?o X y la matriz de vectores de respuesta
#Y. Las caracter?sticas (columnas de X ) est?n dadas por pares de
#mutaci?n (posici?n). Defina  [X_ {i, j} = {1 si el i ?simo paciente
#tiene el  j  ?simo par mutaci?n (posici?n) y 0 en caso contrario} , Y_ {i, j} =
#{resistencia del paciente  i  a la droga j }.] Por ejemplo, en la
#muestra de f?rmacos de tipo PI, se observan tres mutaciones diferentes (A, C y D)
#en la posici?n 63 de la proteasa, por lo que tres columnas de X ( denominadas
#P63.A, P63 .C y P63.D) indican la presencia o ausencia de cada mutaci?n en
#esta posici?n.

# Flatten (aplanar) a matrix to a vector with names from concatenating
# row/column names.
flatten_matrix <- function(M, sep='.') {
  x <- c(M)
  names(x) <- c(outer(rownames(M), colnames(M),
                      function(...) paste(..., sep=sep)))
  x
}

# Construct preliminary design matrix.
muts = c(LETTERS, 'i', 'd')
X = outer(muts, as.matrix(gene_df[,pos_cols]), Vectorize(grepl))
X = aperm(X, c(2,3,1))
dimnames(X)[[3]] <- muts
X = t(apply(X, 1, flatten_matrix))
mode(X) <- 'numeric'

# Remove any mutation/position pairs that never appear in the data.
X = X[,colSums(X) != 0]
# Extract response matrix.
Y = gene_df[,4:(pos_start-1)]
hist(Y[,1], breaks='FD')
hist(log(Y[,1]), breaks='FD')
#Uso del filtro knockoff
knockoff_and_bhq <- function (X, y, q) {
  # Log-transform the drug resistance measurements.
  y = log(y)
  # Remove patients with missing measurements.
  missing = is.na(y)
  y = y[!missing]
```

```
  X = X[!missing,]
  # Remove predictors that appear less than 3 times.
  #Elimina los predictores que aparecen menos de 3 veces.
  X = X[,colSums(X) >= 3]
  # Remove duplicate predictors.
  X = X[,colSums(abs(cor(X)-1) < 1e-4) == 1]
  # Run the knockoff filter.
  knock.gen = function(x) create.fixed(x, method='equi')
  result = knockoff.filter(X, y, fdr=fdr, knockoffs=knock.gen, statistic=
      stat.glmnet_lambdasmax)
  knockoff_selected = names(result$selected)

  # Run BHq.
  p = ncol(X)
  lm.fit = lm(y ~ X - 1) # no intercept
  p.values = coef(summary(lm.fit))[,4]
  cutoff = max(c(0, which(sort(p.values) <= fdr * (1:p) / p)))
  bhq_selected = names(which(p.values <= fdr * cutoff / p))
  list(Knockoff = knockoff_selected, BHq = bhq_selected)
}
fdr = 0.20
results = lapply(Y, function(y) knockoff_and_bhq(X, y, fdr))
print(results[1])
get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)
comparisons <- lapply(results, function(drug) {
  lapply(drug, function(selected) {
    positions = unique(get_position(selected)) # remove possible
        duplicates
    discoveries = length(positions)
    false_discoveries = length(setdiff(positions, tsm_df$Position))
    list(true_discoveries = discoveries - false_discoveries,
         false_discoveries = false_discoveries,
         fdp = false_discoveries / max(1, discoveries))
  })
})
print(comparisons[1])
for (drug in names(comparisons)) {
  plot_data = do.call(cbind, comparisons[[drug]])
  plot_data = plot_data[c('true_discoveries','false_discoveries'),]
  barplot(as.matrix(plot_data), main = paste('Resistance to', drug),
          col = c('navy','orange'), ylim = c(0,40))
}

#PI(APV)

#tomar la variable X e y.
  y1 = log(Y[,1])
  missing = is.na(y1)
  y1 = y1[!missing]
  X_PI_APV = X[!missing,]
  X_PI_APV = X_PI_APV[,colSums(X_PI_APV) >= 3]
  X_PI_APV = X_PI_APV[,colSums(abs(cor(X_PI_APV)-1) < 1e-4) == 1]
dim(X_PI_APV)
```

```
length(y1)

#knockoff
knock.gen = function(x) create.fixed(x, method='equi')
fdr=0.2
get_position <- function(x)
  sapply(regmatches(x, regexpr('[[:digit:]]+', x)), as.numeric)
result1.1 = knockoff.filter(X_PI_APV, y1, fdr=fdr, knockoffs=knock.gen,
                            statistic=stat.glmnet_lambdasmax)
positions1.1=unique(get_position(names(result1.1$selected)))
discoveries1.1 = length(positions1.1)
false_discoveries1.1 = length(setdiff(positions1.1, tsm_df$Position))
true_discoveries1.1 = discoveries1.1 - false_discoveries1.1
fdr1=false_discoveries1.1/discoveries1.1

#B&H
p1 = ncol(X_PI_APV)
lm.fit1 = lm(y1 ~ X_PI_APV - 1) # no intercept
p.values1 = coef(summary(lm.fit1))[,4]
cutoff1 = max(c(0, which(sort(p.values1) <= fdr * (1:p1) / p1)))
bhq_selected1 = names(which(p.values1 <= fdr * cutoff1 / p1))
positions2.1=unique(get_position(bhq_selected1))
discoveries2.1 = length(positions2.1)
false_discoveries2.1 = length(setdiff(positions2.1, tsm_df$Position))
true_discoveries2.1 = discoveries2.1 - false_discoveries2.1
fdr2=false_discoveries2.1/discoveries2.1

#Lassso
q1=cv.glmnet(X_PI_APV,y1)
L1=q1$lambda.min
mlasso1 <- glmnet(
  X_PI_APV          ,
  y1           ,
  lambda      = L1,
  standardize = TRUE
)
which(mlasso1$beta>0)
positions3.1=unique(get_position(colnames(X_PI_APV)[which(mlasso1$beta>0)
   ]))
discoveries3.1 = length(positions3.1)
false_discoveries3.1 = length(setdiff(positions3.1, tsm_df$Position))
true_discoveries3.1 = discoveries3.1 - false_discoveries3.1
fdr3=false_discoveries3.1/discoveries3.1

#Stepwise
p1=dim(X_PI_APV)[1]
d1 = prepare_data(y1, X_PI_APV)
fs1=fast_forward(d1,maxf = p1)
fs1$model
positions4.1=unique(get_position(fs1$model))
discoveries4.1 = length(positions4.1)
false_discoveries4.1 = length(setdiff(positions4.1, tsm_df$Position))
true_discoveries4.1 = discoveries4.1 - false_discoveries4.1
fdr4=false_discoveries4.1/discoveries4.1
```

```
173
174 #Graficas
175 knockoof1=c(true_discoveries1.1,false_discoveries1.1)
176 BHq1=c(true_discoveries2.1,false_discoveries2.1)
177 Lasso1=c(true_discoveries3.1,false_discoveries3.1)
178 Stepwise1=c(true_discoveries4.1,false_discoveries4.1)
179 tabla1=data.frame(knockoof1,BHq1,Lasso1,Stepwise1)
180 barplot(as.matrix(tabla1),
181         main ="APV",
182         xlab = "Metodo",
183         col = c('navy','orange'), ylim = c(0,40))
184
185 #ATV
186
187 #Variables
188 y2 = log(Y[,2])
189 missing2 = is.na(y2)
190 y2 = y2[!missing2]
191 X_PI_ATV = X[!missing2,]
192 X_PI_ATV = X_PI_ATV[,colSums(X_PI_ATV) >= 3]
193 X_PI_ATV = X_PI_ATV[,colSums(abs(cor(X_PI_ATV)-1) < 1e-4) == 1]
194 dim(X_PI_ATV)
195 length(y2)
196
197 #knockoof
198 result2 = knockoff.filter(X_PI_ATV, y2, fdr=fdr, knockoffs=knock.gen,
199                           statistic=stat.glmnet_lambdasmax)
200 positions1.2=unique(get_position(names(result2$selected)))
201 discoveries1.2 = length(positions1.2)
202 false_discoveries1.2 = length(setdiff(positions1.2, tsm_df$Position))
203 true_discoveries1.2 = discoveries1.2 - false_discoveries1.2
204
205 #B&H
206 p2 = ncol(X_PI_ATV)
207 lm.fit2 = lm(y2 ~ X_PI_ATV - 1) # no intercept
208 p.values2 = coef(summary(lm.fit2))[,4]
209 cutoff2 = max(c(0, which(sort(p.values2) <= fdr * (1:p2) / p2)))
210 bhq_selected2 = names(which(p.values2 <= fdr * cutoff2 / p2))
211 positions2.2=unique(get_position(bhq_selected2))
212 discoveries2.2 = length(positions2.2)
213 false_discoveries2.2 = length(setdiff(positions2.2, tsm_df$Position))
214 true_discoveries2.2 = discoveries2.2 - false_discoveries2.2
215
216 #Lassso
217 q2=cv.glmnet(X_PI_ATV,y2)
218 L2=q2$lambda.min
219 mlasso2 <- glmnet(
220   X_PI_ATV          ,
221   y2           ,
222   lambda      = L2,
223   standardize = TRUE
224 )
225 which(mlasso2$beta>0)
226 positions3.2=unique(get_position(colnames(X_PI_ATV)[which(mlasso2$beta>0)
```

```
   ]))
discoveries3.2 = length(positions3.2)
false_discoveries3.2 = length(setdiff(positions3.2, tsm_df$Position))
true_discoveries3.2 = discoveries3.2 - false_discoveries3.2

#Stepwise
p2=dim(X_PI_ATV)[1]
d2= prepare_data(y2, X_PI_ATV)
fs2=fast_forward(d2,maxf = p2)
fs2$model
positions4.2=unique(get_position(fs2$model))
discoveries4.2 = length(positions4.2)
false_discoveries4.2 = length(setdiff(positions4.2, tsm_df$Position))
true_discoveries4.2 = discoveries4.2 - false_discoveries4.2

#Graficas
knockoof2=c(true_discoveries1.2,false_discoveries1.2)
BHq2=c(true_discoveries2.2,false_discoveries2.2)
Lasso2=c(true_discoveries3.2,false_discoveries3.2)
Stepwise2=c(true_discoveries4.2,false_discoveries4.2)
tabla2=data.frame(knockoof2,BHq2,Lasso2,Stepwise2)
barplot(as.matrix(tabla2),
        main = "ATV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,50))

#IDV

#Variables
y3 = log(Y[,3])
missing3 = is.na(y3)
y3 = y3[!missing3]
X_PI_IDV = X[!missing3,]
X_PI_IDV = X_PI_IDV[,colSums(X_PI_IDV) >= 3]
X_PI_IDV = X_PI_IDV[,colSums(abs(cor(X_PI_IDV)-1) < 1e-4) == 1]
dim(X_PI_IDV)
length(y3)

#knockoof
result3 = knockoff.filter(X_PI_IDV, y3, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.3=unique(get_position(names(result3$selected)))
discoveries1.3 = length(positions1.3)
false_discoveries1.3 = length(setdiff(positions1.3, tsm_df$Position))
true_discoveries1.3 = discoveries1.3 - false_discoveries1.3

#B&H
p3 = ncol(X_PI_IDV)
lm.fit3 = lm(y3 ~ X_PI_IDV - 1) # no intercept
p.values3 = coef(summary(lm.fit3))[,4]
cutoff3 = max(c(0, which(sort(p.values3) <= fdr * (1:p3) / p3)))
bhq_selected3 = names(which(p.values3 <= fdr * cutoff3 / p3))
positions2.3=unique(get_position(bhq_selected3))
discoveries2.3 = length(positions2.3)
```

```
false_discoveries2.3 = length(setdiff(positions2.3, tsm_df$Position))
true_discoveries2.3 = discoveries2.3 - false_discoveries2.3

#Lassso
q3=cv.glmnet(X_PI_IDV,y3)
L3=q3$lambda.min
mlasso3 <- glmnet(
  X_PI_IDV      ,
  y3            ,
  lambda      = L3,
  standardize = TRUE
)
which(mlasso3$beta>0)
positions3.3=unique(get_position(colnames(X_PI_IDV)[which(mlasso3$beta>0)
   ]))
discoveries3.3 = length(positions3.3)
false_discoveries3.3 = length(setdiff(positions3.3, tsm_df$Position))
true_discoveries3.3 = discoveries3.3 - false_discoveries3.3

#Stepwise
p3=dim(X_PI_IDV)[1]
d3= prepare_data(y3, X_PI_IDV)
fs3=fast_forward(d3,maxf = p3)
fs3$model
positions4.3=unique(get_position(fs3$model))
discoveries4.3 = length(positions4.3)
false_discoveries4.3 = length(setdiff(positions4.3, tsm_df$Position))
true_discoveries4.3 = discoveries4.3 - false_discoveries4.3

#Graficas
knockoof3=c(true_discoveries1.3,false_discoveries1.3)
BHq3=c(true_discoveries2.3,false_discoveries2.3)
Lasso3=c(true_discoveries3.3,false_discoveries3.3)
Stepwise3=c(true_discoveries4.3,false_discoveries4.3)
tabla3=data.frame(knockoof3,BHq3,Lasso3,Stepwise3)
barplot(as.matrix(tabla3),
        main = "IDV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,50))

#LPV

#Seleccion de variables
y4 = log(Y[,4])
missing4 = is.na(y4)
y4 = y4[!missing4]
X_PI_LPV = X[!missing4,]
X_PI_LPV = X_PI_LPV[,colSums(X_PI_LPV) >= 3]
X_PI_LPV = X_PI_LPV[,colSums(abs(cor(X_PI_LPV)-1) < 1e-4) == 1]
dim(X_PI_LPV)
length(y4)

#knockoof
result4 = knockoff.filter(X_PI_LPV, y4, fdr=fdr, knockoffs=knock.gen,
```

```
                        statistic=stat.glmnet_lambdasmax)
positions1.4=unique(get_position(names(result4$selected)))
discoveries1.4 = length(positions1.4)
false_discoveries1.4 = length(setdiff(positions1.4, tsm_df$Position))
true_discoveries1.4 = discoveries1.4 - false_discoveries1.4

#B&H
p4 = ncol(X_PI_LPV)
lm.fit4 = lm(y4 ~ X_PI_LPV - 1) # no intercept
p.values4 = coef(summary(lm.fit4))[,4]
cutoff4 = max(c(0, which(sort(p.values4) <= fdr * (1:p4) / p4)))
bhq_selected4 = names(which(p.values4 <= fdr * cutoff4 / p4))
positions2.4=unique(get_position(bhq_selected4))
discoveries2.4 = length(positions2.4)
false_discoveries2.4 = length(setdiff(positions2.4, tsm_df$Position))
true_discoveries2.4 = discoveries2.4 - false_discoveries2.4

#Lassso
q4=cv.glmnet(X_PI_LPV,y4)
L4=q4$lambda.min
mlasso4 <- glmnet(
  X_PI_LPV        ,
  y4          ,
  lambda      = L4,
  standardize = TRUE
)
which(mlasso4$beta>0)
positions3.4=unique(get_position(colnames(X_PI_LPV)[which(mlasso4$beta>0)
   ]))
discoveries3.4 = length(positions3.4)
false_discoveries3.4 = length(setdiff(positions3.4, tsm_df$Position))
true_discoveries3.4 = discoveries3.4 - false_discoveries3.4

#Stepwise
p4=dim(X_PI_LPV)[1]
d4= prepare_data(y4, X_PI_LPV)
fs4=fast_forward(d4,maxf = p4)
fs4$model
positions4.4=unique(get_position(fs4$model))
discoveries4.4 = length(positions4.4)
false_discoveries4.4 = length(setdiff(positions4.4, tsm_df$Position))
true_discoveries4.4 = discoveries4.4 - false_discoveries4.4

#Graficas
knockoof4=c(true_discoveries1.4,false_discoveries1.4)
BHq4=c(true_discoveries2.4,false_discoveries2.4)
Lasso4=c(true_discoveries3.4,false_discoveries3.4)
Stepwise4=c(true_discoveries4.4,false_discoveries4.4)
tabla4=data.frame(knockoof4,BHq4,Lasso4,Stepwise4)
barplot(as.matrix(tabla4),
        main = "LPV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,45))

```

```
#NFV

#Seleccion de variables
y5 = log(Y[,5])
missing5 = is.na(y5)
y5 = y5[!missing5]
X_PI_NFV = X[!missing5,]
X_PI_NFV = X_PI_NFV[,colSums(X_PI_NFV) >= 3]
X_PI_NFV = X_PI_NFV[,colSums(abs(cor(X_PI_NFV)-1) < 1e-4) == 1]
dim(X_PI_NFV)
length(y5)

#knockoof
result5 = knockoff.filter(X_PI_NFV, y5, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.5=unique(get_position(names(result5$selected)))
discoveries1.5 = length(positions1.5)
false_discoveries1.5 = length(setdiff(positions1.5, tsm_df$Position))
true_discoveries1.5 = discoveries1.5 - false_discoveries1.5

#B&H
p5 = ncol(X_PI_NFV)
lm.fit5 = lm(y5 ~ X_PI_NFV - 1) # no intercept
p.values5 = coef(summary(lm.fit5))[,4]
cutoff5 = max(c(0, which(sort(p.values5) <= fdr * (1:p5) / p5)))
bhq_selected5 = names(which(p.values5 <= fdr * cutoff5 / p5))
positions2.5=unique(get_position(bhq_selected5))
discoveries2.5 = length(positions2.5)
false_discoveries2.5 = length(setdiff(positions2.5, tsm_df$Position))
true_discoveries2.5 = discoveries2.5 - false_discoveries2.5

#Lassso
q5=cv.glmnet(X_PI_NFV,y5)
L5=q5$lambda.min
mlasso5 <- glmnet(
  X_PI_NFV          ,
  y5           ,
  lambda     = L5,
  standardize = TRUE
)
which(mlasso5$beta>0)
positions3.5=unique(get_position(colnames(X_PI_NFV)[which(mlasso5$beta>0)
   ]))
discoveries3.5 = length(positions3.5)
false_discoveries3.5 = length(setdiff(positions3.5, tsm_df$Position))
true_discoveries3.5 = discoveries3.5 - false_discoveries3.5

#Stepwise
p5=dim(X_PI_NFV)[1]
d5= prepare_data(y5, X_PI_NFV)
fs5=fast_forward(d5,maxf = p5)
fs5$model
positions4.5=unique(get_position(fs5$model))
discoveries4.5 = length(positions4.5)
```

```
false_discoveries4.5 = length(setdiff(positions4.5, tsm_df$Position))
true_discoveries4.5 = discoveries4.5 - false_discoveries4.5

#Graficas
knockoof5=c(true_discoveries1.5,false_discoveries1.5)
BHq5=c(true_discoveries2.5,false_discoveries2.5)
Lasso5=c(true_discoveries3.5,false_discoveries3.5)
Stepwise5=c(true_discoveries4.5,false_discoveries4.5)
tabla5=data.frame(knockoof5,BHq5,Lasso5,Stepwise5)
barplot(as.matrix(tabla5),
        main = "NPV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,50))

#RTV

#Seleccion de variables
y6 = log(Y[,6])
missing6 = is.na(y6)
y6 = y6[!missing6]
X_PI_RTV = X[!missing6,]
X_PI_RTV = X_PI_RTV[,colSums(X_PI_RTV) >= 3]
X_PI_RTV = X_PI_RTV[,colSums(abs(cor(X_PI_RTV)-1) < 1e-4) == 1]
dim(X_PI_RTV)
length(y6)

#knockoof
result6 = knockoff.filter(X_PI_RTV, y6, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.6=unique(get_position(names(result6$selected)))
discoveries1.6 = length(positions1.6)
false_discoveries1.6 = length(setdiff(positions1.6, tsm_df$Position))
true_discoveries1.6 = discoveries1.6 - false_discoveries1.6

#B&H
p6 = ncol(X_PI_RTV)
lm.fit6 = lm(y6 ~ X_PI_RTV - 1) # no intercept
p.values6 = coef(summary(lm.fit6))[,4]
cutoff6 = max(c(0, which(sort(p.values6) <= fdr * (1:p6) / p6)))
bhq_selected6 = names(which(p.values6 <= fdr * cutoff6 / p6))
positions2.6=unique(get_position(bhq_selected6))
discoveries2.6 = length(positions2.6)
false_discoveries2.6 = length(setdiff(positions2.6, tsm_df$Position))
true_discoveries2.6 = discoveries2.6 - false_discoveries2.6

#Lassso
q6=cv.glmnet(X_PI_RTV,y6)
L6=q6$lambda.min
mlasso6 <- glmnet(
  X_PI_RTV          ,
  y6           ,
  lambda      = L6,
  standardize = TRUE
)
```

```
which(mlasso6$beta>0)
positions3.6=unique(get_position(colnames(X_PI_RTV)[which(mlasso6$beta>0)
   ]))
discoveries3.6 = length(positions3.6)
false_discoveries3.6 = length(setdiff(positions3.6, tsm_df$Position))
true_discoveries3.6 = discoveries3.6 - false_discoveries3.6

#Stepwise
p6=dim(X_PI_RTV)[1]
d6= prepare_data(y6, X_PI_RTV)
fs6=fast_forward(d6,maxf = p6)
fs6$model
positions4.6=unique(get_position(fs6$model))
discoveries4.6 = length(positions4.6)
false_discoveries4.6 = length(setdiff(positions4.6, tsm_df$Position))
true_discoveries4.6 = discoveries4.6 - false_discoveries4.6

#Graficas
knockoof6=c(true_discoveries1.6,false_discoveries1.6)
BHq6=c(true_discoveries2.6,false_discoveries2.6)
Lasso6=c(true_discoveries3.6,false_discoveries3.6)
Stepwise6=c(true_discoveries4.6,false_discoveries4.6)
tabla6=data.frame(knockoof6,BHq6,Lasso6,Stepwise6)
barplot(as.matrix(tabla6),
        main = "RTV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,45))

#SQV

#Seleccion de la variable
y7 = log(Y[,7])
missing7 = is.na(y7)
y7 = y7[!missing7]
X_PI_SQV = X[!missing7,]
X_PI_SQV = X_PI_SQV[,colSums(X_PI_SQV) >= 3]
X_PI_SQV = X_PI_SQV[,colSums(abs(cor(X_PI_SQV)-1) < 1e-4) == 1]
dim(X_PI_SQV)
length(y7)

#knockoof
result7 = knockoff.filter(X_PI_SQV, y7, fdr=fdr, knockoffs=knock.gen,
                          statistic=stat.glmnet_lambdasmax)
positions1.7=unique(get_position(names(result7$selected)))
discoveries1.7 = length(positions1.7)
false_discoveries1.7 = length(setdiff(positions1.7, tsm_df$Position))
true_discoveries1.7 = discoveries1.7 - false_discoveries1.7

#B&H
p7 = ncol(X_PI_SQV)
lm.fit7 = lm(y7 ~ X_PI_SQV - 1) # no intercept
p.values7 = coef(summary(lm.fit7))[,4]
cutoff7 = max(c(0, which(sort(p.values7) <= fdr * (1:p7) / p7)))
bhq_selected7 = names(which(p.values7 <= fdr * cutoff7 / p7))
```

```
positions2.7=unique(get_position(bhq_selected7))
discoveries2.7 = length(positions2.7)
false_discoveries2.7 = length(setdiff(positions2.7, tsm_df$Position))
true_discoveries2.7 = discoveries2.7 - false_discoveries2.7

#Lassso
q7=cv.glmnet(X_PI_SQV,y7)
L7=q7$lambda.min
mlasso7 <- glmnet(
  X_PI_SQV      ,
  y7           ,
  lambda      = L7,
  standardize = TRUE
)
which(mlasso7$beta>0)
positions3.7=unique(get_position(colnames(X_PI_SQV)[which(mlasso7$beta>0)
   ]))
discoveries3.7 = length(positions3.7)
false_discoveries3.7 = length(setdiff(positions3.7, tsm_df$Position))
true_discoveries3.7 = discoveries3.7 - false_discoveries3.7

#Stepwise
p7=dim(X_PI_SQV)[1]
d7= prepare_data(y7, X_PI_SQV)
fs7=fast_forward(d7,maxf = p7)
fs7$model
positions4.7=unique(get_position(fs7$model))
discoveries4.7 = length(positions4.7)
false_discoveries4.7 = length(setdiff(positions4.7, tsm_df$Position))
true_discoveries4.7 = discoveries4.7 - false_discoveries4.7

#Graficas
knockoof7=c(true_discoveries1.7,false_discoveries1.7)
BHq7=c(true_discoveries2.7,false_discoveries2.7)
Lasso7=c(true_discoveries3.7,false_discoveries3.7)
Stepwise7=c(true_discoveries4.7,false_discoveries4.7)
tabla7=data.frame(knockoof7,BHq7,Lasso7,Stepwise7)
barplot(as.matrix(tabla7),
        main = "SQV",
        xlab = "Metodo",
        col = c('navy','orange'), ylim = c(0,40))
```

Listing B.33: PI

Bibliografía

[Barber and Candès, 2015] Barber, R. F. and Candès, E. J. (2015). Controlling the false discovery rate via knockoffs. *The Annals of Statistics*, 43(5):2055–2085.

[Barber and Candès, 2019] Barber, R. F. and Candès, E. J. (2019). A knockoff filter for high-dimensional selective inference. *Annals of Statistics*, 47(5):2504–2537.

[Benjamini and Hochberg, 1995] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300.

[Casella and Berger, 2002] Casella, G. and Berger, R. (2002). R. 2001, statistical inference. *Duxbury Press*.

[Draper and Smith, 1998] Draper, N. R. and Smith, H. (1998). *Applied regression analysis*, volume 326. John Wiley & Sons.

[Efron, 2012] Efron, B. (2012). *Large-scale inference, empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press.

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

[Gelman and Carlin, 2014] Gelman, A. and Carlin, J. (2014). Beyond power calculations, assessing type s (sign) and type m (magnitude) errors. *Perspectives on Psychological Science*, 9(6):641–651.

[Gelman and Tuerlinckx, 2000] Gelman, A. and Tuerlinckx, F. (2000). Type s error rates for classical and bayesian single and multiple comparison procedures. *Computational Statistics*, 15(3):373–390.

[González et al., 2015] González, A. et al. (2015). Selección de variables: Una revisión de métodos existentes. *Universidad da Coruña*.

[Hastie et al., 2015] Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical learning with sparsity: the lasso and generalizations*. CRC press.

[Hochberg, 1988] Hochberg, Y. (1988). A sharper bonferroni procedure for multiple tests of significance. *Biometrika*, 75(4):800–802.

[Hoerl and Kennard, 1970] Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

[Holm, 1979] Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70.

[Hommel, 1988] Hommel, G. (1988). A stagewise rejective multiple test procedure based on a modified bonferroni test. *Biometrika*, 75(2):383–386.

[Lu et al., 2019] Lu, J., Qiu, Y., and Deng, A. (2019). A note on type s/m errors in hypothesis testing. *British Journal of Mathematical and Statistical Psychology*, 72(1):1–17.

[Rhee et al., 2005] Rhee, S.-Y., Fessel, W. J., Zolopa, A. R., Hurley, L., Liu, T., Taylor, J., Nguyen, D. P., Slome, S., Klein, D., Horberg, M., et al. (2005). Hiv-1 protease and reverse-transcriptase mutations: correlations with antiretroviral therapy in subtype b isolates and implications for drug-resistance surveillance. *The Journal of infectious diseases*, 192(3):456–465.

[Simes, 1986] Simes, R. J. (1986). An improved bonferroni procedure for multiple tests of significance. *Biometrika*, 73(3):751–754.

Alojamiento de la Tesis en el Repositorio Institucional	
Título de Tesis:	SELECCIÓN DE VARIABLES EN MODELOS DE REGRESIÓN LINEAL CONTROLANDO LA TASA DE FALSOS POSITIVOS.
Autor(a) o autores(ras) de la Tesis:	Lic. Leonardo Alfonso Martínez González
ORCID:	0009-0007-1144-8133
Resumen de la Tesis:	En regresión lineal un problema de interés es la selección de las variables explicativas que verdaderamente están relacionadas con la variable de respuesta en el modelo lineal. Dos métodos clásicos de selección de variables son Stepwise y Lasso, y otro método propuesto más recientemente por Barber y Candès en 2015 es el Knockoff, que busca controlar la tasa de falsos positivos. En este trabajo se analiza el desempeño del método Knockoff para seleccionar variables de un modelo de regresión lineal controlando la tasa de falsos positivos. El análisis se hace mediante simulaciones y se compara el desempeño de este método con Stepwise y Lasso, en términos de la tasa de falsos positivos y descubrimientos verdaderos. Se presenta también un ejemplo de aplicación a datos reales encontrados en la literatura.
Palabras claves de la Tesis:	Knockoff, Lasso, falsos positivos, selección de variables, regresión lineal.
Referencias citadas:	Barber, R. F., & Candès, E. J. (2015). Controlling the false discovery rate via knockoffs. *The Annals of Statistics, 43*(5), 2055–2085. https://doi.org/10.1214/15-AOS1337 Barber, R. F., & Candès, E. J. (2019). A knockoff filter for high-dimensional selective inference. *Annals of Statistics, 47*(5), 2504–2537. https://doi.org/10.1214/18-AOS1711 Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological), 57*(1), 289–300. Casella, G., & Berger, R. (2002). Statistical inference (2nd ed.). Duxbury Press.

Draper, N. R., & Smith, H. (1998). *Applied regression analysis* (Vol. 326). John Wiley & Sons.

Efron, B. (2012). *Large-scale inference: Empirical Bayes methods for estimation, testing, and prediction* (Vol. 1). Cambridge University Press.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer Series in Statistics.

Gelman, A., & Carlin, J. (2014). Beyond power calculations: Assessing type S (sign) and type M (magnitude) errors. *Perspectives on Psychological Science, 9*(6), 641–651. https://doi.org/10.1177/1745691614551642

Gelman, A., & Tuerlinckx, F. (2000). Type S error rates for classical and Bayesian single and multiple comparison procedures. *Computational Statistics, 15*(3), 373–390.

González, A., et al. (2015). Selección de variables: Una revisión de métodos existentes. *Universidad da Coruña.*

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). *Statistical learning with sparsity: The lasso and generalizations.* CRC Press.

Hochberg, Y. (1988). A sharper Bonferroni procedure for multiple tests of significance. *Biometrika, 75*(4), 800–802.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics, 12*(1), 55–67.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics, 6*(2), 65–70.

Hommel, G. (1988). A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika, 75*(2), 383–386.

Lu, J., Qiu, Y., & Deng, A. (2019). A note on type S/M errors in hypothesis testing. *British Journal of Mathematical and Statistical Psychology, 72*(1), 1–17.

Rhee, S.-Y., Fessel, W. J., Zolopa, A. R., Hurley, L., Liu, T., Taylor, J., Nguyen, D. P., Slome, S., Klein, D., Horberg, M., et al. (2005). HIV-1 protease and reverse-transcriptase mutations: Correlations with antiretroviral therapy in subtype B isolates and implications for drug-resistance surveillance. *The Journal of Infectious Diseases, 192*(3), 456–465.

Simes, R. J. (1986). An improved Bonferroni procedure for multiple tests of significance. *Biometrika, 73*(3), 751–754.