Lec3: Regression

Isidoro Garcia Urquieta

2023

Agenda

- Ajuste de Bonferroni (Holm - Bonferroni)
- Holm Bonferroni vs FDR
- Aprendizaje Estadístico
- Regresión Lineal

Repaso FDR

Recordemos que la alta dimensionalidad de (n, p) en Big data hace que:

- Rechazemos muchas hipótesis nulas ($H_0 : \beta = 0$): Falsos positivos.

Repaso FDR

Recordemos que la alta dimensionalidad de (n, p) en Big data hace que:

- Rechazemos muchas hipótesis nulas ($H_0 : \beta = 0$): Falsos positivos.
- La clase pasada vimos como el *FDR* castiga a los p-values de acuerdo a su ranking para, con base en una q de falsos positivos, encontrar un punto de corte óptimo: $p^*(q) = max\{p : p(k) \leq k\frac{q}{N}\}$

Repaso FDR

Recordemos que la alta dimensionalidad de (n, p) en Big data hace que:

- Rechazemos muchas hipótesis nulas ($H_0 : \beta = 0$): Falsos positivos.
- La clase pasada vimos como el *FDR* castiga a los p-values de acuerdo a su ranking para, con base en una q de falsos positivos, encontrar un punto de corte óptimo: $p^*(q) = max\{p : p(k) \leq k\frac{q}{N}\}$
- Hoy veremos dos métodos de correción de pruebas de hipótesis múltiples nuevos.

Bonferroni

Si tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

El ajuste de Bonferroni corrige los falsos positivos al fijar un corte muy agresivo:

$$p^* = \frac{\alpha}{m}$$

Por ejemplo, si tenemos una base con 20 columnas, con una $\alpha = 0.05$, fijas $p^* = \frac{0.05}{20} = 0.00125$.

Que diferencias ven con FDR. cual parece mejor?

Holm Bonferroni

El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor

Holm Bonferroni

El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor
- Es $P(1) < \frac{\alpha}{m}$? si? Rechaza H_{01}. no? FIN del procedimiento

Holm Bonferroni

El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor
- Es $P(1) < \frac{\alpha}{m}$? si? Rechaza H_{01}. no? FIN del procedimiento
- Es $P(2) < \frac{\alpha}{m-1}$? si? Rechaza H_{02}. no? FIN del procedimiento

Holm Bonferroni

El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor
- Es $P(1) < \frac{\alpha}{m}$? si? Rechaza H_{01}. no? FIN del procedimiento
- Es $P(2) < \frac{\alpha}{m-1}$? si? Rechaza H_{02}. no? FIN del procedimiento
- ...

Holm Bonferroni

El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor
- Es $P(1) < \frac{\alpha}{m}$? si? Rechaza H_{01}. no? FIN del procedimiento
- Es $P(2) < \frac{\alpha}{m-1}$? si? Rechaza H_{02}. no? FIN del procedimiento
- ...
- Es $P(k) < \frac{\alpha}{m+1-k}$? si? Rechaza H_{0k}. no? FIN del procedimiento

Holm Bonferroni

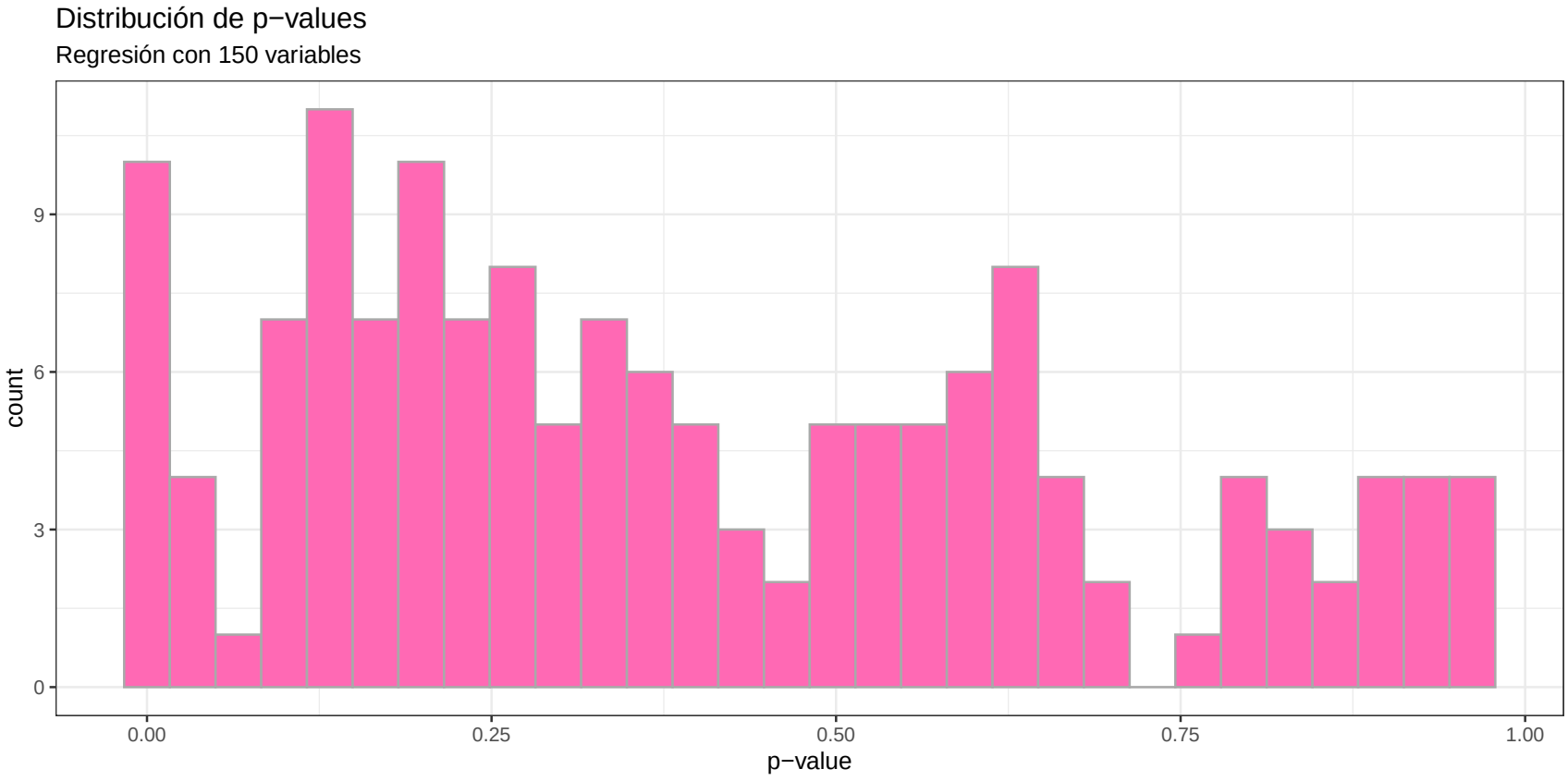
El procedimiento de Holm Bonferroni es una alternativa a Bonferroni que es menos estricta. Se ve de la siguiente manera:

Tenemos $H_{01}, H_{02}, H_{03}, ..., H_{0m} : \beta_{i \in m} = 0$ pruebas de hipótesis con sus p-values $p_{01}, p_{02}, p_{03}, ..., p_p$

- Ordenas los p-values de menor a mayor
- Es $P(1) < \frac{\alpha}{m}$? si? Rechaza H_{01}. no? FIN del procedimiento
- Es $P(2) < \frac{\alpha}{m-1}$? si? Rechaza H_{02}. no? FIN del procedimiento
- ...
- Es $P(k) < \frac{\alpha}{m+1-k}$? si? Rechaza H_{0k}. no? FIN del procedimiento
- Noten como el ajuste de Holm-Bonferroni es mas relajado que el Bonferroni. Con mayor formalidad, el ajuste de bonferroni controla demasiado por el error tipo I dejando espacio a muchos falsos negativos (error tipo II). El ajuste de Holm-Bonferroni permite un poco de más error tipo I pero disminuye mucho el error tipo II.

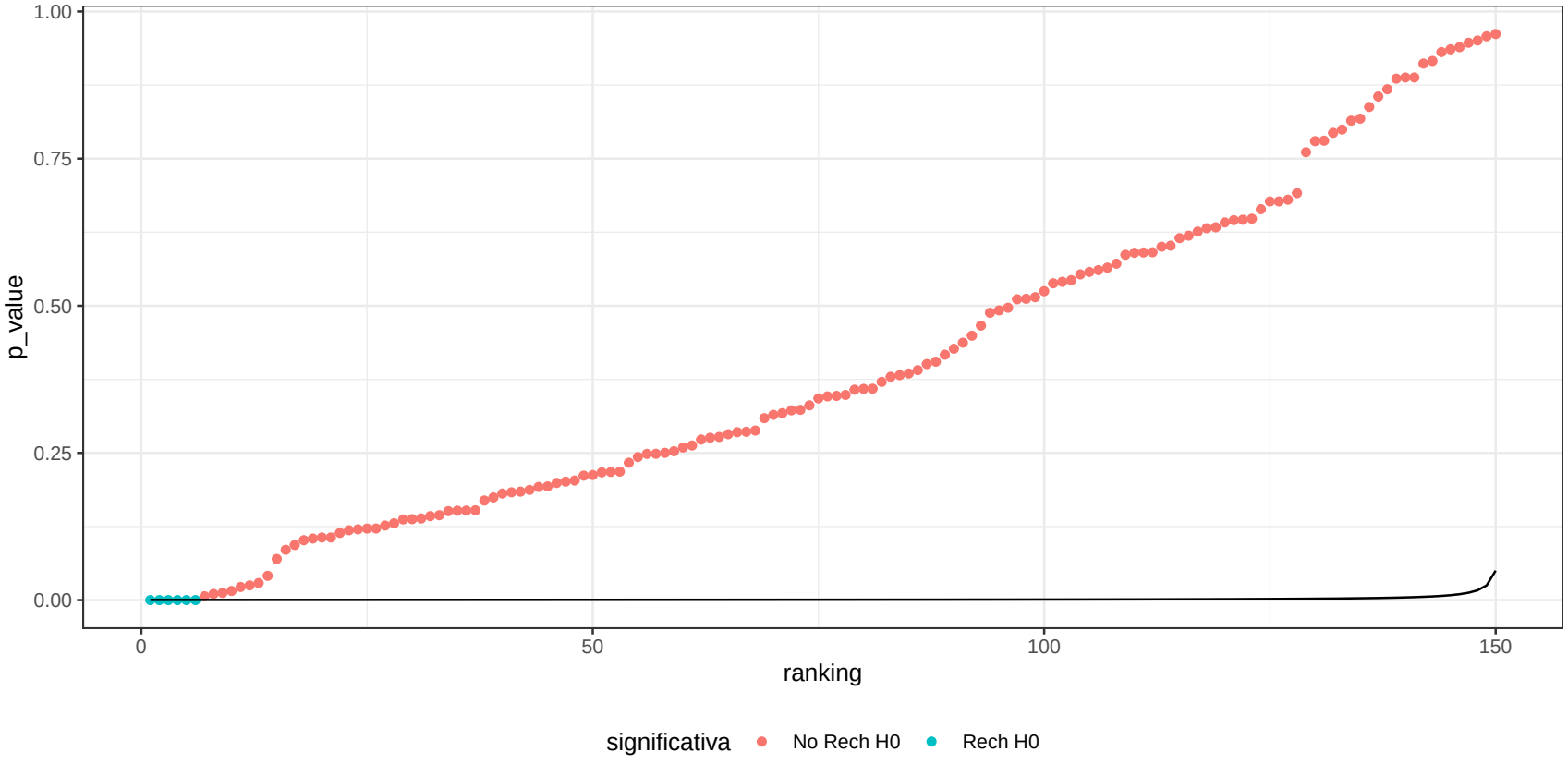
Ejemplo Holm + Bonferroni

Veamos la distribución de 150 p-values en una regresión.



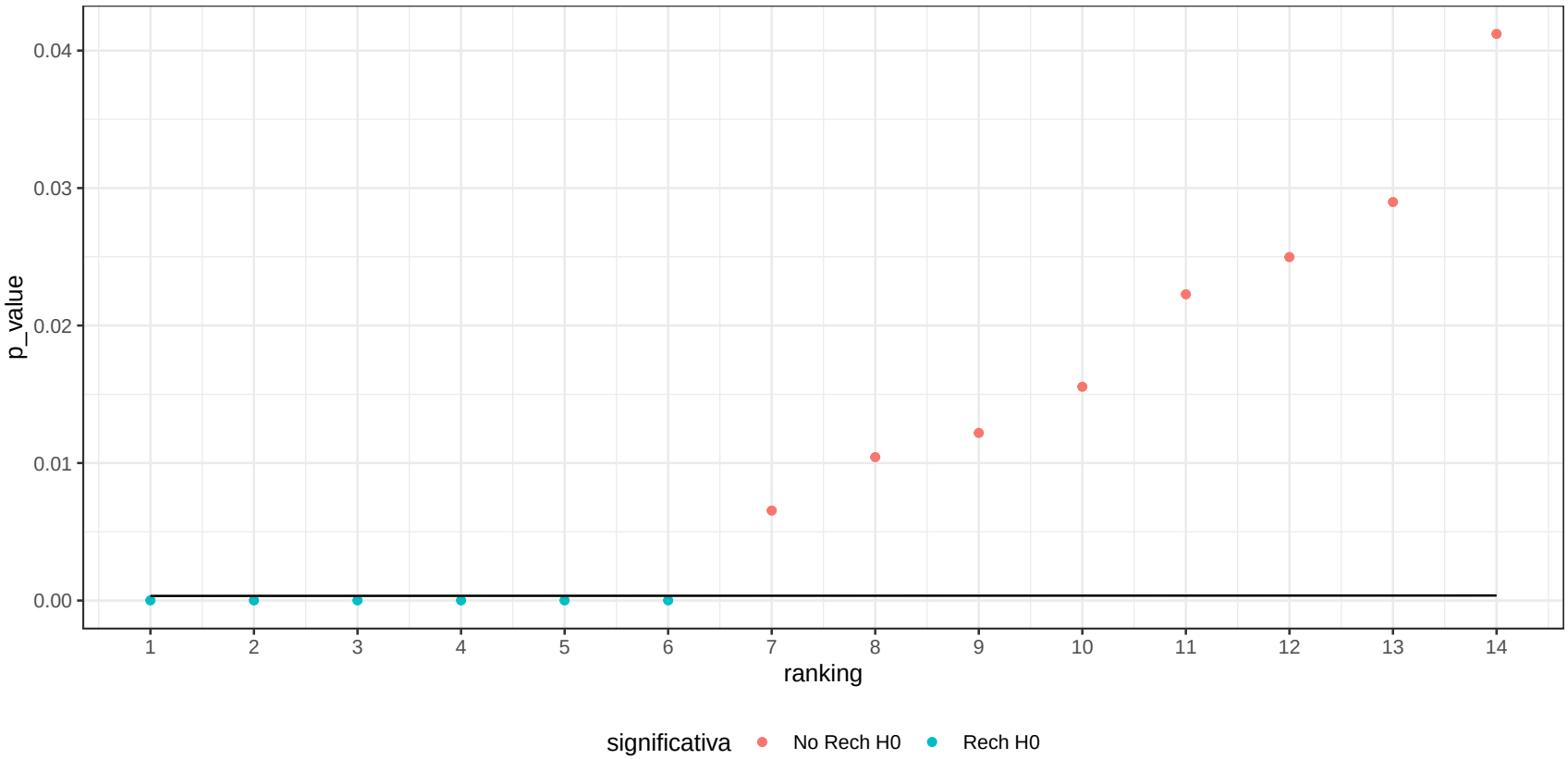
Ejemplo Holm + Bonferrioni

Veamos como se ve la regla $P(k) < \frac{\alpha}{m+1-k}$ graficamente:



Ejemplo Holm + Bonferroni

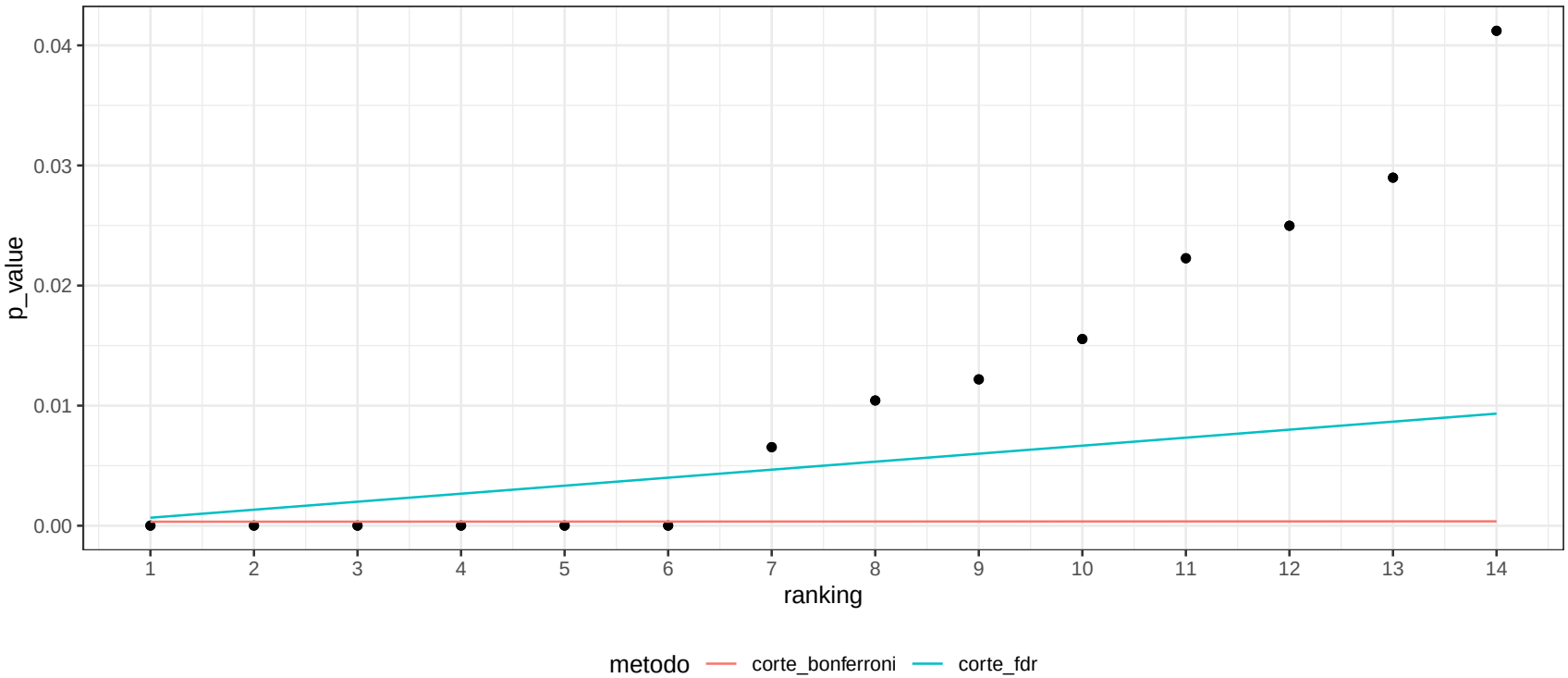
Si hacemos zoom-in a valores que hubieran sido significativos bajo la inferencia 'normal' (i.e. $< .05$)




```
## [1] 0.00001
```

Comparativa vs FDR

Como se compara FDR (a $q = 0.1$) con Holm + Bonferroni? Veamoslo graficamente para el mismo ejemplo:



FDR es menos astringente de Holm Bonferroni. Esto hace que *FDR* sea típicamente preferido sobre Holm Bonferroni en casos de Big Data.

Inferencia en Big Data resumen

Ante una gran dimensión de datos (i.e. muchas pruebas y muchas filas en la base), tenemos que estar muy alertas de los falsos positivos. Los métodos de inferencia estadística se quedan cortos ante esto.

Vimos dos métodos muy importantes para el control de falsos positivos ante pruebas múltiples:

- **False Discovery Rate**: Estimador del False Discovery Proportion. Necesitas elegir una tasa de falsos positivos máxima deseada (q).
  - Pros: Balancea bien entre los falsos positivos y los falsos negativos.
  - Cons: Asume cierta independencia entre las pruebas.
- **Holm Bonferroni**: Regla de decisión sencilla.
  - Pros: No necesitas asignar una tasa máxima de falsos positivos deseados. No asume nada sobre la relación entre las pruebas.
  - Cons: Es muy astringente. Esto puede generar muy poco error tipo I pero más error tipo II.

Aprendizaje Estadístico

El aprendizaje estadístico concierne dos tipos:

1) **Aprendizaje Supervisado**: Existe una variable endógena que gobierna el aprendizaje. Esto genera la siguiente función a estimar:

$$Y = f(X) + \epsilon$$

Donde $X : X_1, ..., X_p$ son todas las variables explicativas que se nos ocurran, Y es la variable supervisora y ϵ es un error aleatorio.

2) **Aprendizaje No Supervisado**: No existe una variable objectivo. Tenemos X y queremos entender como se relacionan entre ellas. Típicamente relacionado a método de reducción de dimensionalidad.

Empecemos con aprendizaje supervisado.

Apredizaje Supervisado

El aprendizaje estadístico (incluido el Machine Learning) tratan de estimar f. La(s) variable(s) Y que supervisan el modelo dictan si se cumplió el objetivo del modelo.

Existen dos objetivos primordiales para estimar $f(.)$:

1) Predicción: Generamos $\hat{Y} = \hat{f}(X)$ como estimador que predice Y.

Donde $\hat{f}$ es nuestro estimador de f y $\hat{Y}$ es nuestra predicción de Y. ϵ es el error. Donde $E[\epsilon] = 0$. Típicamente no estamos muy interesados en la forma de f. Únicamente nos interesa su poder predictivo.

2) Inferencia: Estamos interesados en entender la relación de Y con $X_1, ..., X_p$. Esto es, cómo cambia f cuando cambia alguna X. Por ende, en estos casos debemos conocer la forma de $f(.)$.

Cómo estimamos f

Vamos a aprender a estimar f de manera **paramétrica** y **no paramétrica**.

Paramétrica

- Asumimos una forma funcional (i.e. $f(X) = \beta_0 + \beta_1 X_1 + ...\beta_p X_p$).
- Estimamos los parámetros de la función $\beta_0, \beta_1, ...\beta_p$.

Pros:

- Es un enfoque sencillo. Reduces el problema a un vector a estimar.
- Interpretabilidad: Es muy fácil entender f.

Cons:

- La elección de $f(X)$. Si la elección que hacemos de $f(X)$ difiere mucho de la realidad, podemos tener un modelo que sea muy malo. Esto es especialmente cierto en un mundo de Big Data donde la cantidad de variables no permite exploración 2x2 de variables.

Cómo estimamos f

No paramétrica

- No asumes ninguna forma funcional. Ajustas los datos con estadísticos en submuestras (i.e. medias en un árbol).

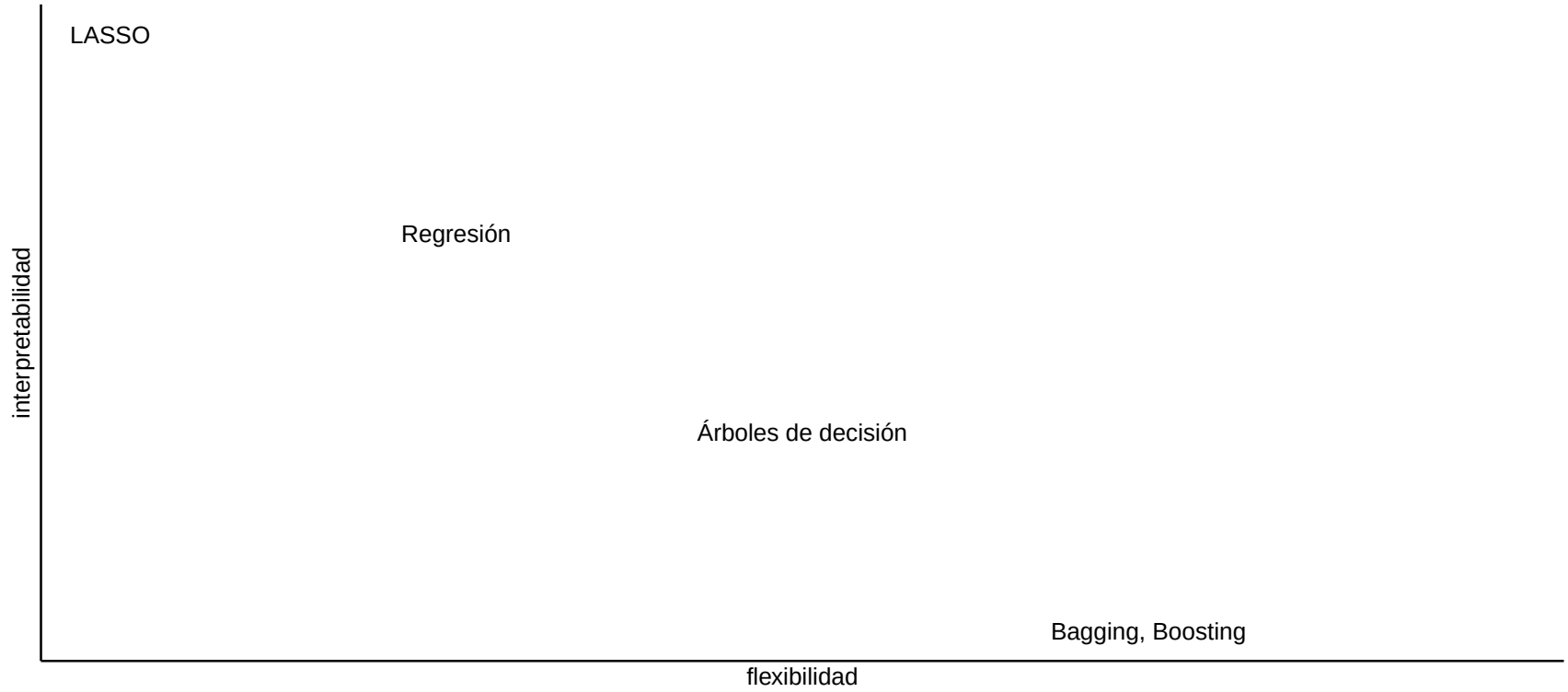
Pros:

- Al no hacer ningún supuesto sobre $f(X)$, es más probable que la estimación se parezca a los datos. Son más flexibles.

Cons:

- El problema continúa siendo complejo. Típicamente los algoritmos no parámetricos son más inestables a nuevos datos y requieren de bases mucho más grande para hacer estimaciones precisas.

Trade-off Flexibilidad vs Interpretabilidad



Como elegimos el método de estimación de f?

Hasta ahora vimos que existen algoritmos supervisados y no supervisados. En ambos, queremos estimar $f(X)$ de manera paramétrica y no paramétrica. Sin embargo como elegimos entre los distintos métodos de aprendizaje estadístico.

Poder predictivo: En el caso de los algoritmos predictivos, estamos interesados en algoritmos que sean buenos prediciendo datos que no hayan visto. A esto le llamamos su desempeño fuera de la muestra (OOS por Out of Sample).

Pasos:

1. Divides de manera aleatoria tu base en 2: Entrenamiento $((y, x_1, ..., x_p)^T)$ y Validación $((y, x_1, ..., x_p)^{(N-T)})$.
2. Estimas $f(X)$ en la base de entrenamiento.
3. Predecimos $\hat{Y}$ en la base de validación. Medimos el desempeño predictivo del modelo.

In-sample vs Out-of-sample

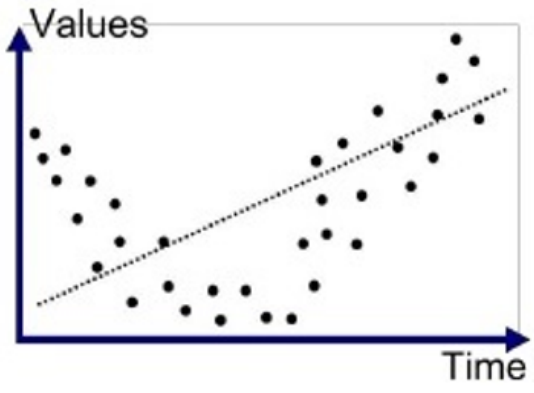
Noten como esta es la primera diferencia vs la Econometría clásica. En un mundo de Big data, tenemos suficientes datos para dividirlos en dos partes y así estimar el mejor modelo posible.

La creación de la base de validación nos permite no caer en **overfitting**. Esto es, no crear un modelo que ajuste demasiado a los datos en la base de entrenamiento tal que **siga ruido**.

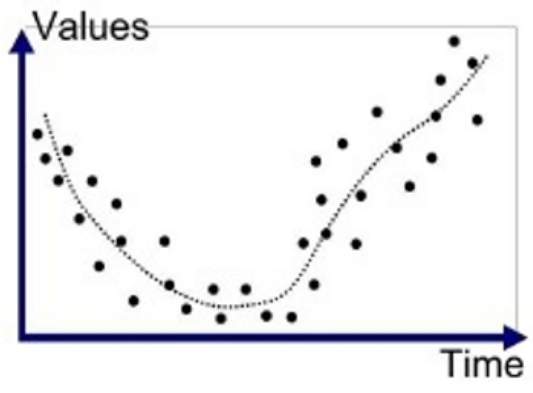
Lo anterior genera un problema de predicción, pues el modelo no será útil para predecir datos que no vio en el entrenamiento. Así, hay un trade-off entre la complejidad del modelo y su poder predictivo.

Por otro lado, un modelo que sea demasiado sencillo (**underfitted**), no explica bien la realidad. Este tampoco va a predecir bien fuera de la muestra.

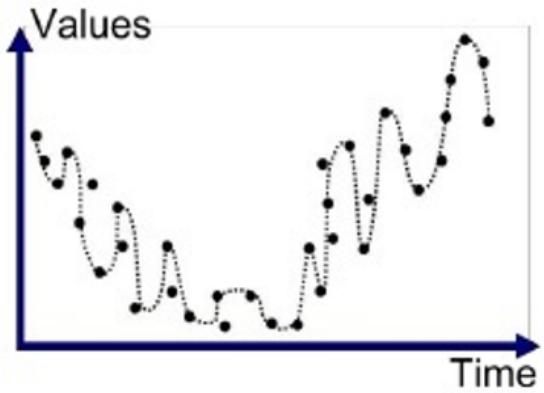
Ejemplo Overfitting



Underfitted



Good Fit/Robust



Overfitted

Regresión Lineal

- Este es el primer método de predicción e inferencia que veremos. Muchos de los algoritmos de Machine Learning nacen de la regresión lineal. Así que es importante tener un entendimiento correcto de su funcionamiento.

La regresión lineal poblacional se puede expresar con la siguiente ecuación:

$$E[y|x] = f(x'\beta)$$

$$y_i = f(x'\beta) + u_i$$

Donde $f(x'\beta)$ es una función **lineal en parámetros** y u_i es el error. x es una matriz de dimensión (n, k) y β es el vector de parámetros a estimar k.

Estimación

La regresión lineal se puede estimar mediante:

1) Mínimos Cuadrados Ordinarios ($min_{\{\beta\}} \Sigma_{i=1}^{N} (y_i - \bar{y})^2$) o
2) Máxima Verosimilitud ($max_{\{\beta\}} \Sigma_{i=1}^{N} ln(f(x_i|\beta))$

Ambas dan resultados iguales para el caso más simple. Para casos complejos (i.e. Logit!) sólo se puede usar Máxima verosimilitud.

El estimador para cualquier X_k es:

$$\hat{\beta}_k = \frac{\Sigma_i^N (x_i - \bar{x}) y_i}{\Sigma_i^N (x_i - \bar{x})^2}$$

Si $n- > \infty$ $\hat{\beta}_k$ converge en probabilidad a:

$$\hat{\beta}_k \rightarrow^p = \frac{Cov(x_i, y_i)}{Var(x_i)}$$

Supuestos Regresión Lineal

Recordemos los supuestos de Gauss-Markov:

1. La regresión es lineal en parámetros: Ojo! Lineal en parámetros no significa lineal a las X's. Podemos ajustar muchos tipos de función, siempre y cuando sea lineal en parámteros (i.e. no β^2).
2. Muestreo aleatorio (Independencia): Este supuesto es necesario para que las $\hat{\beta}$ sean variables aleatorias.
3. $Var(X) \neq 0$: Hay varianza en las $X's$. Sino los estimadores se vuelven infinito. Además, no debe haber perfecta multicolinealidad.
4. $E[u|x] = 0$: Controlando por las X's, el error se promedia a cero. Exogeneidad
5. $Var[u|x] = \sigma^2$: La varianza de u es constante para cualquier valor de las X's. Esto es homocedasticidad.
6. $y|x \sim N(x'\beta, \sigma^2)$: Normalidad. Este es opcional. Para grandes números no es necesario.

Desviaciones de los supuesto - implicaciones

Ahora veamos que significa algunos de los supuestos en palabras simples y sus desviaciones:

1. Linealidad
2. Podemos aplicar inferencia estadística. Violaciones: errores persistentes! (i.e. Series de tiempo).
3. El estimador de cada β_i es un valor no infinito $\hat{\beta}_k = \frac{Cov(x_i, y_i)}{Var(x)}$. Una violación es poner dos variables X demasiado correlacionadas (multicolinealidad).
4. Estimamos un modelo que incluye todas las variables relevantes. Fallas a este supuesto se dan cuando una variable que es relevante y relacionada con alguna X no se incluyen (lo que genera sesgo).
5. Violaciones: Errores tipo clúster (geograficos, panel!).

Implicaciones de los supuestos: BLUE

Si estos supuestos se cumplen, resulta que la regresión es un estimador BLUE (Best Linear Unbiased Estimator). Esto es, que las $\hat{\beta}$ tendrán la mínima varianza (y ECM) dentro de todos los estimadores.

Vamos a probarlo. Para ello, recordemos algunas cosas importantes para simplificar:

Una propiedad de MCO es que $E[\epsilon_i|x] = 0$ (*Exogeneidad*)

La covarianza es un operador lineal.

La covarianza entre una constante y una variable es 0.

La covarianza entre X y X es la varianza de X.

La Ley de las Esperanzas Iteradas: $E[yx] = E[XE[y|X]]$

BLUE

$$Y = X\beta + u$$

$$\hat{\beta} = (X'X)^{-1}X'Y$$

$$\hat{\beta} = (X'X)^{-1}X'(X\beta + u)$$

$$\hat{\beta} = (X'X)^{-1}X'X\beta + (X'X)^{-1}X'u$$

$$\hat{\beta} = \beta + (X'X)^{-1}X'u$$

Esto significa que $\hat{\beta}_k$ es una variable aleatoria (contiene a $u \sim N(0, 1)$).

BLUE

Si es BLUE, saquemos su $E[\hat{\beta}]$ y su $Var[\hat{\beta}]$:

$$E[\hat{\beta}] = \beta + E[(X'X)^{-1}X'u]$$
$$Var[\hat{\beta}] = Var[(X'X)^{-1}X'u]$$

Usamos la ley de esperanzas iteradas $E[zx] = E[xE[z|x]]$

$$E[\hat{\beta}] = \beta + E[(X'X)^{-1}X'E[u|x]] = \beta$$
$$Var[\hat{\beta}] = Var[(X'X)^{-1}X'u] = (X'X)^{-1}X'Var[u|x] = (X'X)^{-1}\sigma^2$$

Noten como asumí que la $Cov(u, X) = 0$. Es decir, asumí que no hay ommited variable bias.

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

Factibilidad de supuestos

Típicamente, es muy díficil argumentar que estimamos el modelo *real* o *correcto* que permita que el estimador de MCO sea insesgado o causal ($Cov(x, u) = 0$. Es fácil argumentar, sobretodo con pocas variables, que existe alguna variable que sea *relevante* y *endogena a* T.

En el mundo del Big Data, tendríamos muchas variables. Con esto, podemos llegar a argumentar que no hay variables omitidas. Sin embargo, ahora debemos incluir todas las variables???? No! recuerden que la varianza de los estimadores crece y crece ante la inclusión de ruido.

En la econometría clásica, vamos a la teoría económica para conocer el modelo 'real' de Y. Sin embargo, en muchos casos estos modelos han resultado ser demasiado simples (underfitted).

Otra manera es hacer stepwise. En un mundo de muchas variables (Big data), esto podría ser un proceso demasiado tardado.

Stepwise

Veamos todos los modelos posibles para 10 y 100 variables:

$$(\frac{100}{1}) + (\frac{100}{2}) + (\frac{100}{3}) + ..., (\frac{100}{100})$$

```
# Combinaciones de modelos para 10 variables
numeros<-seq(1,10)
total<-max(numeros)
(combinaciones<-sum(map_dbl(numeros, ~choose(n = total, k = .))))
```

```
## [1] 1023
```

```
# Para 100
numeros<-seq(1,100)
total<-max(numeros)
(combinaciones<-sum(map_dbl(numeros, ~choose(n = total, k = .))))
```

```
## [1] 1267650600228229401496703205375
```

Selección de la estimación

Vean como el número creció exponencialmente. Para 100 variables, las combinaciones posibles son casi infitas.

A pesar de lo poderosa que es la regresión lineal, necesitamos alguna manera más moderna de hacer selección de variables (Próxima clase!).

Regresión lineal como predictor

Hasta ahora cubrimos los supuestos de Gauss Markov y las implicaciones que puede tener en un mundo de big data para hacer inferencia causal.

Para la predicción, la regresión también puede ser muy poderosa. Recordemos cual sería la métrica (OOS) para medir el desempeño del modelo:

$$ECM = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Recuerden que estamos usando que:

$$y_i = \hat{y}_i + u_i$$

Se ve familiar para el contexto de Econometría?

R^2

La suma total de los cuadrados. Es la variabilidad total de y:

$$TSS = \Sigma_{i=1}^{N}(y_i - \bar{y})^2 = ESS + RSS$$

La suma cuadrada de residuales. Es cuanto no explica el modelo.

$$RSS = \Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2 = \Sigma_{i=1}^{N} u_i^2 = N * ECM$$

La suma cuadrada de lo explicado. Que tan lejos va nuestro modelo de la media:

$$ESS = \Sigma_{i=1}^{N}(\hat{y}_i - \bar{y})^2$$

Definamos la R^2 o bondad de ajuste:

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

R^2

La R^2 se interpreta como la proporción de la variabilidad total que se explica con el modelo. Noten como en el contexto de predicción es simétrico. Si baja RSS, baja el error de predicción y sube la R^2.

Entonces, como usamos la regresión como predictor? Metemos todas las variables hasta subir R^2 a 1? No!

Recuerden que si uno estima un modelo muy complejo (i.e. muchos controles), es probable que logre que suba la R^2 in-sample, pero el error de predicción fuera de la muestra sea muy alto.

Otra vez estamos ante un problema de elegir cual regresión correr...

Selección de modelos

La primera comparativa entre modelos debe ser mi modelo de regresión vs el modelo nulo (i.e. sólo sacar la media de y sin controles).

$$H_0 : \beta_1 = \beta_2 = \beta_3 = 0 \qquad H_a : \beta_i \neq 0$$

El estadístico que se usa es:

$$F = \frac{(TSS - RSS)/k}{RSS/(n - k - 1)}$$

Noten como estoy viendo si la suma de cuadrados explicados por mi modelo supera significativamente a la suma de errores al cuadrado.

Selección de modelos (2)

La segunda comparativa es un modelo vs otro modelo:

Modelo 1: $y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + u_i$

Modelo 2: $y_i = \beta_0 + \beta_1 x_1 + u_i$

El estadistico de prueba es:

$$F = \frac{(RSS_1 - RSS_0)/q}{RSS_0/(n-p-1)}$$

Este estadístico compara los errores cuadrados de un modelo vs el otro de manera proporcional. Si se gana suficiente R^2 con la adición de una o muchas variables, el modelo es mejor.

Noten como todas estas comparaciones se tendrían que hacer OOS.

No-linealidades en X

Finalmente, el supuesto 1 de la regresión indica la linealidad en parámetros. Esto no nos impide agregar funciones más complejas de X:

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + u_i$$

Otro gran ejemplo es que podemos partir el espacio con interacciones para detectar no linealidades adicionales

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + u_i$$

Resumen regresión

- Método parámetrico relativamente fácil de estimar (via MCO o MV)
- La linealidad lo hace muy fácil de interpretar
- Si incluimos el modelo correcto (i.e. estructural), la regresión nos arroja estimadores insesgados/causales.
- En este problema, la estimación del modelo correcto es **crucial**. Con alta dimensionalidad, este problema se sale de las manos. Necesitamos algoritmos de selección de variables.
- La regresión también puede ser muy útil para la predicción. Nos enfrentamos al problema elegir el modelo correcto.
- Finalmente, la regresión te permite estimar $f(.)$ muy complejas. Sin embargo, es muy fácil caer en overfitting. (que interacción agregar? cuáles variables elevadas a potencias?)
- La solución a estos problemas es la selección de variables vía regularización. Esto lo veremos la siguiente clase.

Regresion en `R`

La libreria principal para correr regresiones en `R` es `stats::lm`. La librería `stats` viene pre cargada. `lm` es por 'linear model'.

Para correr regresiones con datos tipo panel se puede usar:

- `lfe::felm`: Es la mejor libreria. Te deja incluir efectos fijos, variables instrumentales y errores clusterizados. Lamentablemente la libreria salió de CRAN de manera transitoria por un error.
- `fixest::feols`: Este paquete `library(fixest)` es una gran librería de regresiones con efectos fijos y errores clusterizados. Es más parecida a `reghdfe` en STATA por su rapidez.

Regresión en R

```
library(lfe)
library(fixest)
library(broom)
diamonds<-diamonds

regresion<-lm(carat~depth+table+x+z+y, data = diamonds)

tidy(regresion)
```

```
## # A tibble: 6 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -2.77     0.0281      -98.8  0
## 2 depth        0.0180   0.000367     49.1  0
## 3 table        0.00165  0.000210      7.84 4.65e-15
## 4 x            0.396    0.00239     166.   0
## 5 z            0.00470  0.00302       1.56 1.20e- 1
## 6 y            0.0130   0.00174       7.47 8.06e-14
```

Regresión en R

```
summary(regresion)
```

```
##
## Call:
## lm(formula = carat ~ depth + table + x + z + y, data = diamon
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7234 -0.0714 -0.0333  0.0595  3.7925
##
## Coefficients:
##               Estimate Std. Error t value             Pr(>|t|
## (Intercept) -2.7717638  0.0280602 -98.779 < 0.000000000000000
## depth        0.0180281  0.0003669  49.140 < 0.000000000000000
## table        0.0016461  0.0002100   7.838  0.000000000000046
## x            0.3962228  0.0023875 165.959 < 0.000000000000000
## z            0.0046963  0.0030185   1.556                0.1
## y            0.0129853  0.0017380   7.471  0.000000000000805
## ---
```

Logit

La regresión logística es muy útil para los casos donde Y es binaria (i.e. 0 = false o 1= true).

Algunos ejemplos:

- La persona va a hacer default del crédito que le dimos?
- Esta persona va a comprar el producto que le mostramos.

Recordemos que la regresión lineal se basa en $E[y|x] = f(X'\beta)$. Cuando y es binaria, esto se convierte en:

$$E[y|x] = p(y = 1|x) * 1 + p(y = 0|x) * 0 = p(y = 1|x)$$

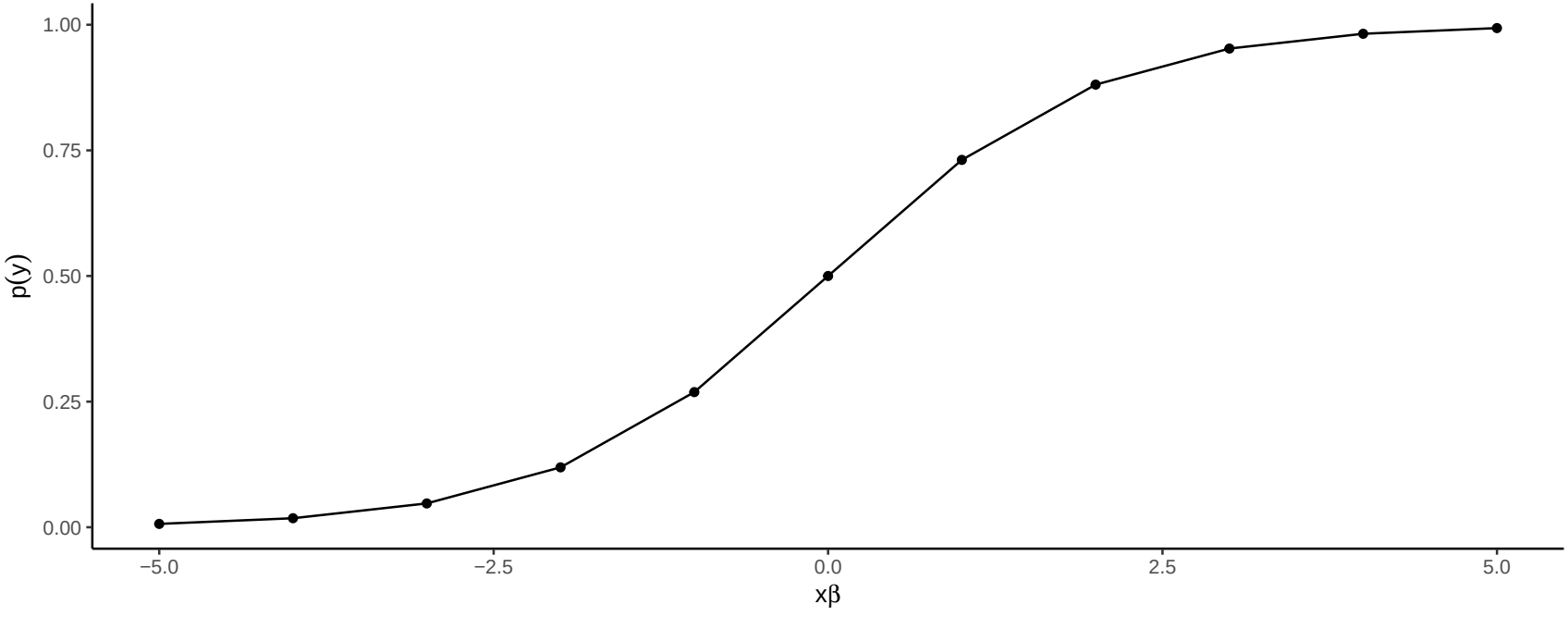
Por lo tanto, estamos modelando:

$$p(y = 1|x) = f(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p)$$

Logit

La regresión logística usa un tipo de distribución:

$$p(y = 1|x) = \frac{e^{x'\beta}}{1 + e^{x'\beta}}$$



Logit

El 'truco' del logit es que modela el ratio de la probabilidad de 1 sobre la probabilidad de cero.

$$log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$$

Logit en R

```
logit<-glm(ideal~x+y+z+depth+table, data = diamonds)
logit
```

```
##
## Call:  glm(formula = ideal ~ x + y + z + depth + table, data
##
## Coefficients:
## (Intercept)            x            y            z          dep
##  12.3223163   -0.0264423    0.0040566    0.0004384   -0.06882
##
## Degrees of Freedom: 53939 Total (i.e. Null);  53934 Residual
## Null Deviance:       12940
## Residual Deviance: 8513  AIC: 53500
```

Logit en R

```
tidy(logit)
```

```
## # A tibble: 6 x 5
##   term          estimate std.error statistic p.value
##   <chr>            <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept) 12.3        0.109     113.      0
## 2 x           -0.0264     0.00930    -2.84    0.00447
## 3 y            0.00406    0.00677     0.599   0.549
## 4 z            0.000438   0.0118      0.0373  0.970
## 5 depth       -0.0688     0.00143   -48.2     0
## 6 table       -0.131      0.000818 -161.      0
```

Logit

Por qué no usar una regresión lineal simple? La respuesta es CUANDO usar uno o otro.

- El Modelo de Probabilidad Lineal es una gran opción cuando queremos comparar entre grupos (i.e. en experimentos). Esto es porque los grupos nunca se saldrán del rango $[0, 1]$ y porque así no hacemos ningún supuesto de la distribución de $p(y = 1|x)$
- El logit es más útil para **modelos**. Es decir, cuando queremos modelar el nivel de la probabilidad. En este caso, no es muy útil un modelo que asegure que la probabilidad no salga de $[0, 1]$.