1. Fundamento para la Red Bayesiana

Para problemas de clasificación una de las posibles soluciones es utilizar una regresión logística, la cual es de la forma $f(x) = \frac{L}{1+e^{-k(x-x_0)}}$. En este caso utilizamos esta regresión logística de manera logarítmica, de esta forma nos queda de la siguiente manera:

$$\log(\frac{P(Y=1)}{1-(P(Y=1))}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 \tag{1}$$

En donde es directo despejar y llegar a que esto es:

$$P(Y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\beta_3 X_3)}} \tag{2}$$

También conocida como logit function, tenemos que $logit(p_i j) = \log(\frac{p}{1-p}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$.

Por otra parte con el teorema de bayes sabemos que:

$$P(Y^{DATA}|\beta) \rightarrow P(\beta|Y^{DATA}) = \frac{P(Y^{DATA}|\beta)P(\beta)}{P(Y^{DATA})} \tag{3}$$

De esta forma buscaremos valor para β como una función de probabilidad, los cuales pertenecerán a la solución de nuestro "logit function", por lo que partiendo de un algoritmo para encontrar estos beta's $P(\beta|Y^{DATA})$ para finalmente utilizarlo de la manera $P(Y^{DATA}|\beta)$.

La cantidad de β's se puede definir de manera arbitraria, por ejemplo podemos usar:

$$logit(P_i j) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 \tag{4a}$$

$$logit(P_i j) = \beta_0 + \beta_1 X_1 + \beta_4 X_1^2 + \beta_2 X_2 + \beta_5 X_2^2 + \beta_3 X_3 + \beta_6 X_3^2 \tag{4b}$$

$$logit(P_i j) = \beta_0 + \beta_1 X_1 + \beta_4 X_1^2 + \beta_7 X_1^3 + \beta_2 X_2 + \beta_5 X_2^2 + \beta_8 X_2^3 + \beta_3 X_3 + \beta_6 X_3^2 + \beta_9 X_3^2 \tag{4c}$$

De esta forma tendremos a cada beta no como un número o un "peso"sino como un función de probabilidad:

$$\begin{aligned} \beta_i &\sim \mathcal{N}(\mu_{\beta_i}, \sigma^2_{beta_i}) \\ logit(P_i j) &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 \\ Y_{ij} &= Bernoulli(P_{ij}) \end{aligned}$$

Siguiendo la idea mostrada en (3) también se puede probar otras formulaciones arbitrarias como la siguiente:

$$P(Y=1|\beta_1, \beta_2, \beta_3, PP, slope, humedad) \tag{5}$$

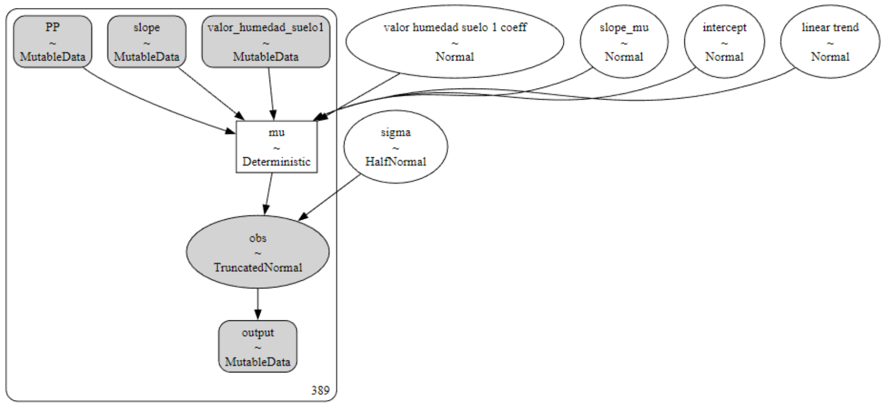

Figura 1: Nodos del modelo

2. Algoritmo Bayesiano y MCMC

Para encontrar $P(\beta_n|y^{data})$ se computa $\beta_1, ..., \beta_n$ Para el calculo se utiliza la computación llamada "Bayesian maximum likelihood":

$$\beta^{mode} = \arg\max_{\beta} \log[p(y^{data|\beta})] + \sum_{i=1}^{N} \log[p_i(\beta_i)] \tag{6}$$

Cuando $N \to \infty$ nuestro algoritmo aproxima $p(\beta|y^{data})$ de manera perfecta

Así tenemos que:

$$f(\beta^{(Nx1)}|Y) = \frac{f(Y|\beta)f(\beta)}{f(Y)} \tag{7}$$

Lo que buscamos es la distribución posterior de β_i:

$$h(\beta_i|y) = \int f(\beta|y)d\beta, (i, ..., N) \tag{8}$$

Donde el algoritmo Metropolis-Hastings (MCMC) nos va a permitir aproximar esta distribución de $h(\beta_i|y)$ este algoritmo selecciona candidatos para $\beta^{(r)}$, y para el calculo de β^{mode} se utiliza el descenso de gradiente(no se mostrará aquí) de la formula (6).

No se va a mostrar cada paso del algoritmo ya que no es el objetivo, pero es importante ver la etapa 1 y 3 de este mismo En la etapa 1 sabemos que se computa $-\beta^{(1)} = \beta^*$, ahora para $\beta^r > 1$ se eligen candidatos para β^r:

$$draw\, x\, from\, \beta^{(r-1)} + \kappa N(0, V) \tag{9}$$

Donde κ será un escalar, y el valor del factor $\kappa N(0, V)$ será cada "salto"que da la distribución Para la etapa 3 el algoritmo computa:

$$\beta^{(r)} = \begin{cases} \beta^{(r)}, & \text{si } \mu > \alpha \\ x, & \text{si } \mu < \alpha \end{cases}$$