(giới thiệu chung mô hình hồi quy tuyến tính)

Hồi quy tuyến tính là dạng hồi quy rất phổ biến hiện nay.

Cách tiếp của phương pháp hồi quy tuyến tính là tìm quan hệ tuyến tính giữa một hoặc nhiều biến phụ thuộc với một hoặc nhiều biến độc lập khác.

Trong chương này ta sẽ xét đến 2 dạng hồi quy tuyến tính là hồi quy tuyến tính đơn giản (Simple Linead Regression) và hồi quy tuyến tính đa biến (multiple linear regression). Chú ý thuật ngữ phân biệt với hồi quy bội (multivariate linear regression) sẽ đề cập ở chương sau.

Simple linear regression:

Chúng ta đi tìm tương quan tuyến tính giữa 2 biến x và y (một biến độc lập và một biến phụ thuộc). Như vậy ta sẽ đi tìm phương trình đường thẳng, càng sát càng tốt, cho phép tính các giá trị dự báo (predict values) của biến phụ thuôc tại các giá trị biến độc lập sát với dữ liệu observed nhất.

Cách đánh giá thế nào là “sát nhất” (best fit) của hồi quy tuyến tính có thể được sử dụng bằng một số phương pháp khác nhau.

Bắt đầu từ phương trình đường thẳng đơn giản:

$$y = \alpha + \beta x$$

Trong đó $$\beta$$ được gọi là hệ số góc và $$\alpha$$ là hệ số cắt intercept.

Để dựng được phương trình quan hệ này chúng ta có được một bộ dữ liệu thực nghiệm observations gồm tập hợp các cặp giá trị {(xi, yi), i = 1, ..., n}.

Ta có thể mô tả lại quan hệ tuyến tính của bộ dữ liệu này với tham gia của 1 giá trị nữa là giá trị sai số (error) hay còn gọi là phần dư (residuals) $$\epsilon\_i$$

$$y\_i = \alpha + \beta x\_i + \epsilon\_i$$

Có thể thấy quan hệ giữa 2 biến này được thể hiện dựa trên 2 hệ số $$\alpha$$ và $$\beta$$ của đường hồi quy

Như đã nói ở trên, việc đánh giá thế nào là sát nhất (best fit) có thể có nhiều phương pháp. Phổ biến nhất là đánh giá dựa trên phương pháp bình phương nhỏ nhất (OLS).

Để hiểu rõ hơn ta có thể bắt đầu với bài toán đơn giản sử dụng bộ dữ liệu mẫu có sẵn trên R

|  |
| --- |
| attach(cars); str(cars) |

|  |
| --- |
| 'data.frame': 50 obs. of 2 variables:  $ speed: num 4 4 7 7 8 9 10 10 10 11 ...  $ dist : num 2 10 4 22 16 10 18 26 34 17 ... |

Đây là bộ dữ liệu này có 2 biến là tốc độ xe speed và khoảng cách dist cần để xe dừng hẳn

Vẽ thử các điểm dữ liệu ta có:

|  |
| --- |
| ds <- cars  library(ggplot2)  ggplot(ds, aes(x = speed, y = dist)) + # Set up canvas with outcome variable on y-axis  geom\_point() # Plot the actual points |



Fig 5.1

Nhiệm vụ của chúng ta là phải tìm quan hệ giữa 2 biến này, ta có công thức đường hồi quy tuyến tính như sau:

$$y\_dist = \alpha\_speed + \beta\_speed x\_speed$$

Để tìm các hệ số hồi quy $$\alpha\_speed$$ và $$\beta\_speed$$ ta thực hiện hồi quy tuyến tính trên R:

fit <- lm(dist ~ speed, ds)

summary(fit)



|  |
| --- |
| Call:  lm(formula = dist ~ speed, data = ds)  Residuals:  Min 1Q Median 3Q Max  -29.069 -9.525 -2.272 9.215 43.201  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) -17.5791 6.7584 -2.601 0.0123 \*  speed 3.9324 0.4155 9.464 1.49e-12 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 15.38 on 48 degrees of freedom  Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438  F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12 |

Trên kết quả hồi quy này ta thấy, phần mềm trả về cho ta hệ số intercept $$\alpha\_dist = -17.58$$ và hệ góc $$\beta\_dist = 3.93$$

Tính các giá trị predict và độ lệch residuals tại các giá trị observed biến độc lập

|  |
| --- |
| ds$predicted <- predict(fit) # Save the predicted values  ds$residuals <- residuals(fit) # Save the residual values |



Fig 5.2

Trên hình ta có thể thấy được đường hồi quy màu xanh là đường thể hiện quan hệ tuyến tính:

$$y\_dist = -17.58 + 3.93 x\_speed$$

Đưa các giá trị $$x\_i$$ vào phương trình ta tính được các giá trị predicted vẽ màu đỏ

Các giá trị này có độ lệch residuals $$\epsilon\_i$$ so với các giá trị thực $$y\_i$$ tương ứng (chấm đen trên hình).

Từ phương trình hồi quy bên trên, ta dễ dàng tính được:

$$\hat{\epsilon\_i} = y\_i - \alpha - \beta x\_i$$

Theo phương pháp bình phương nhỏ nhất, đường best fit là đường có $$\alpha$$ và $$\beta$$ sao cho tổng bình phương độ lệch residuals $$\hat{\epsilon\_i}$$ nhỏ nhất.

Như vậy, đường fit lý tưởng là đường:

$$\sum\_{i=1}^n \epsilon\_i = 0$$

|  |
| --- |
| summary(fit)$r.squared |

|  |
| --- |
| [1] 0.6510794 |

Đánh giá p-value 1.49e-12 < 0.05

Hồi quy tuyến tính Bayesian trên bayesvl:

Cũng với bài toán hồi quy tuyến tính đơn giản này, theo hướng tiếp cận baysian, ta có các phân phối xác xuất giá trị $$y\_{dist}$$:

$$y\_{dist} ~ normal(\mu\_{dist}, \sigma\_{dist})$$

Trong đó:

$$\mu\_{dist} = \alpha\_{speed} + \beta\_{speed} x\_{speed}$$

Giả định ban đầu cho cho các priors:

$$\alpha\_{speed} ~ normal(0, 10)$$

$$\beta\_{speed} ~ normal(0,10)$$

$$\sigma\_{dist} ~ normal(0,10)$$

Ta có thể thấy không còn giá trị sai số $$\epsilon\_i$$ như trong frequentist vì ta đã có biên “độ tin cậy” $$\sigma\_{dist}$$ cho các giá trị $$y\_{dist}$$

|  |
| --- |
| # Design the model  model <- bayesvl()  model <- bvl\_addNode(model, "speed", "norm")  model <- bvl\_addNode(model, "dist", "norm")  model <- bvl\_addArc(model, "speed", "dist", "slope") |

Trên mô hình này ta định nghĩa 2 biến là dist và speed. Quan hệ giữa speed và dist là hồi quy tuyến tính

|  |
| --- |
| # Fit the model  fit <- bvl\_modelFit(model, cars, warmup = 2000, iter = 4000, chains = 4, cores = 1) |

|  |
| --- |
| > summary(fit)  Model Info:  nodes: 2  arcs: 1  scores: NA  formula: dist ~ a\_dist + b\_speed\_dist \* speed  Estimates:  Inference for Stan model: dc3b164adc4fe47a3b8332c0bfb6a401.  4 chains, each with iter=20000; warmup=2000; thin=1;  post-warmup draws per chain=18000, total post-warmup draws=72000.  mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff Rhat  a\_dist -17.39 0.04 7.00 -31.08 -22.05 -17.40 -12.74 -3.58 24553 1  b\_speed\_dist 3.92 0.00 0.43 3.07 3.64 3.92 4.21 4.76 24316 1  Samples were drawn using NUTS(diag\_e) at Tue Mar 10 18:00:17 2020.  For each parameter, n\_eff is a crude measure of effective sample size,  and Rhat is the potential scale reduction factor on split chains (at  convergence, Rhat=1).  elapsed time: 76.9043509960175 secs |

|  |
| --- |
| bvl\_plotParams(fit) |



Fig 5.3

Ta có posteriors hệ số góc b\_speed\_dist và intercept a\_dist

|  |  |
| --- | --- |
|  |  |

Fig 5.4

cắt$$\lpha\lpha{}$$$$\eta\eta{}$$

khoảng tin cậy 80% posterior

Tương tự nếu ta giữ nguyên hệ số góc b=mean(b\_dist)=3.92, cho hệ số cắt intercept chạy trong khoảng tin cậỵ 80% của phân phối posterior hệ số a\_dist ta sẽ có:

|  |  |
| --- | --- |
|  |  |

Fig 5.5

Tổng kết:

|  |
| --- |
| Nếu n quan sát cho ta tập hợp các cặp $$(x\_1, y\_1), $$(x\_2, y\_2), .. , $$(x\_n, y\_n), sai số $$\epsilon\_i ~ N(0, \signma^2)$$. Với các hệ số thực $\alpha$ (intercept) và $\beta$ (slope) ta sẽ có mô hình hồi quy tuyến tính đơn giản sau:  $$y\_i = \alpha + \beta x\_i$$  Mô hình phù hợp (với dữ liệu quan sát) được coi là mô hình thoả mãn:  $$\hat{y}\_i = \hat{\alpha} + \hat{\beta} x\_i$$ |

Hồi quy đa biến:

Hồi quy đa biến tương tự như hồi quy đơn giản ở phần trên nhưng khác là số biến độc lập tăng lên p biến từ 1..p

Khi đó ta có bộ dữ liệu $$\{y\_i, x\_{i1}, ... , x\_{ip}\}\_{i=1}^n$$ với p là số biến độc lập.

$$y\_i = \alpha + \beta\_1 x\_{i1} + \beta\_2 x\_{i2} + ... + \beta\_p x\_{ip} + \epsilon\_i $$

Trong đó:

Số mẫu observed i = 1-n

Thực hành với bộ số liệu 1042:

burden[i] = alpha + beta\_res \* res[i] + beta\_insured \* insured[i]

|  |
| --- |
| # Design the model  model <- bayesvl()  model <- bvl\_addNode(model, "burden", "norm")  model <- bvl\_addNode(model, "res", "norm")  model <- bvl\_addNode(model, "insured", "norm")  model <- bvl\_addArc(model, "res", "burden", "slope")  model <- bvl\_addArc(model, "insured", "burden", "slope") |

|  |
| --- |
| # Generate the stan code for model  model\_string <- bvl\_model2Stan(model)  cat(model\_string) |

|  |
| --- |
| functions{  int numLevels(int[] m) {  int sorted[num\_elements(m)];  int count = 1;  sorted = sort\_asc(m);  for (i in 2:num\_elements(sorted)) {  if (sorted[i] != sorted[i-1])  count = count + 1;  }  return(count);  }  }  data{  // Define variables in data  int<lower=1> Nobs; // Number of observations (an integer)  real burden[Nobs]; // outcome variable  real res[Nobs];  real insured[Nobs];  }  transformed data{  // Define transformed data  }  parameters{  // Define parameters to estimate  real<lower=0> sigma\_burden;  real a\_burden;  real b\_res\_burden;  real b\_insured\_burden;  }  transformed parameters{  // Transform parameters  real mu\_burden[Nobs];  for (i in 1:Nobs) {  mu\_burden[i] = a\_burden + b\_res\_burden \* res[i] + b\_insured\_burden \* insured[i];  }  }  model{  // Priors  a\_burden ~ normal(0,100);  b\_res\_burden ~ normal( 0, 10 );  b\_insured\_burden ~ normal( 0, 10 );  // Likelihoods  burden ~ normal(mu\_burden, sigma\_burden);  }  generated quantities {  // simulate data from the posterior  real yrep\_burden[Nobs];  // log-likelihood posterior  vector[Nobs] log\_lik\_burden;  for (i in 1:num\_elements(yrep\_burden)) {  yrep\_burden[i] = normal\_rng(mu\_burden[i], sigma\_burden);  }  for (i in 1:Nobs) {  log\_lik\_burden[i] = normal\_lpdf(burden[i] | mu\_burden[i], sigma\_burden);  }  } |

|  |
| --- |
| # Fit the model  fit <- bvl\_modelFit(model, data1, warmup = 2000, iter = 20000, chains = 4, cores = 1) |

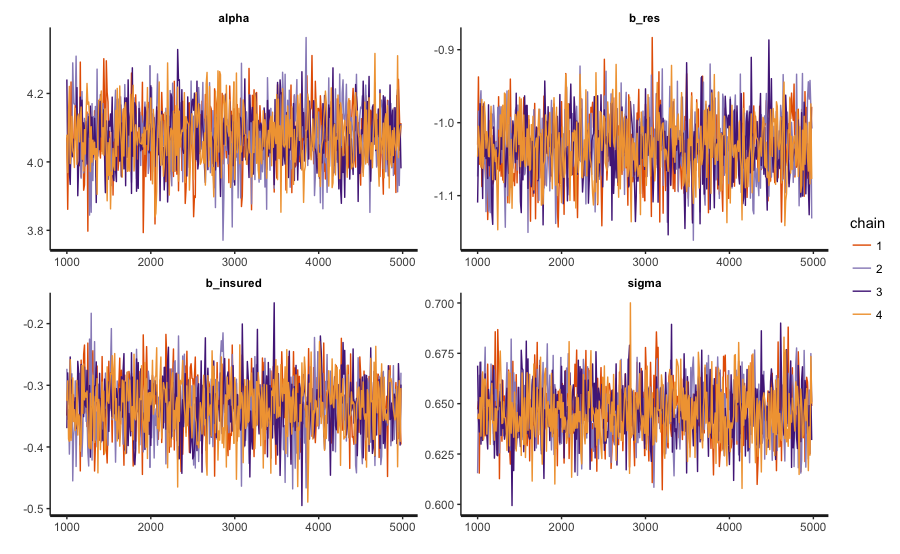
|  |
| --- |
| > summary(model)  4 chains, each with iter=5000; warmup=1000; thin=10;  post-warmup draws per chain=400, total post-warmup draws=1600.  mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff Rhat  alpha 4.08 0 0.09 3.90 4.02 4.08 4.14 4.24 1485 1  b\_res -1.03 0 0.04 -1.12 -1.06 -1.03 -1.01 -0.95 1502 1  b\_insured -0.33 0 0.05 -0.43 -0.37 -0.33 -0.30 -0.24 1610 1  sigma 0.65 0 0.01 0.62 0.64 0.65 0.66 0.67 1763 1 |

Các hệ số hồi quy đều âm, ảnh hưởng burden ngược với res, insured???

burden ~ 4.08 -1.03 \* res -0.33\*insured

res có ảnh hưởng đến finacial burden nhiều hơn insured

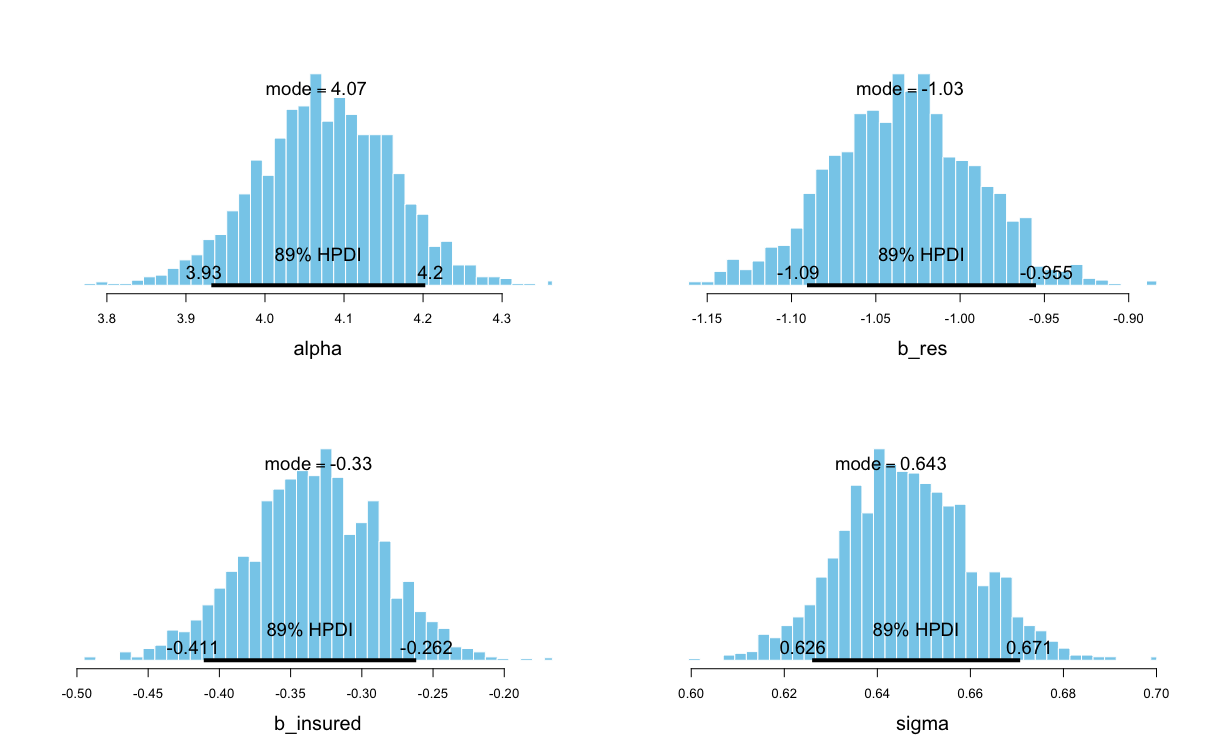
traceplot(fit,pars = c("alpha","b\_res","b\_insured","sigma"))



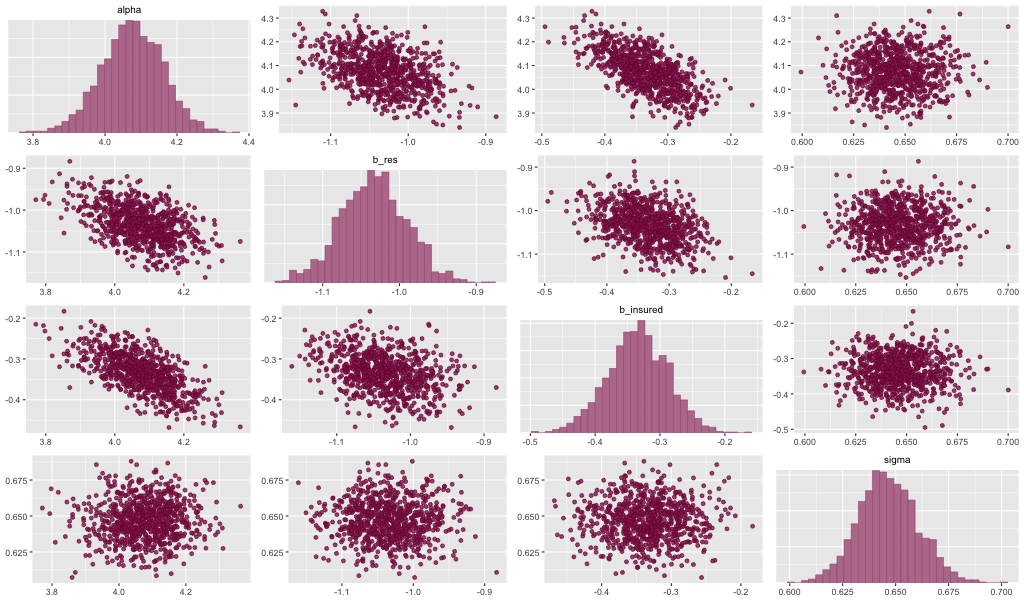
MCMC chain

Các hệ số posteriors:

stan\_hist(fit,pars = params, bins = 50)



Phân phối các hệ số hồi quy



Tương quan các hệ số trong mô hình

Pair tham số:

