December 17, 2017

Dominant/Non-dominant Factors for Insurance Cost & Number of Quotes

By Xiaolong (Shawn) Yang, Pan Corlos Wong, and Melanie Qu

Part I: Purpose

As the "grand old dame of data analysis," the insurance industry is now exploding with vast potential for future data scientists to extract meaningful data. We were interested in insurance data particularly, because analyses of these data ensure more accurate pricing of policies, improve customized product and services, foster stronger customer relationships, and prevent unnecessary financial losses. We sampled 1000 out of 97009 customers from Allstate's Kaggle challenge[1] and extracted various features, including the demographics of the customers and the age of their vehicles. The purpose of our analysis with the sample data is to identify the dominant/non-dominant factors for analyses of insurance costs and number of quotes obtained by the customers before purchasing insurance. The questions we were trying to address are

1. It is commonly accepted that homeowner status (Y/N), marital status (Y/N), and age of vehicle (in Yrs) are dominant factors in determining insurance cost. Is that true with our data?
2. It is logical to think that, the more expensive insurance is, the more number of quotes customers obtain before purchasing insurance. Is that true with our data?

Part II. Data

We used a SQL query to extract the features that we were trying to examine from the large dataset from Kaggle. Then we set a seed and extracted a sample of 1000 data points. The variables, including and not limited to the target and explanatory variables, are included below along with a brief description:

- ***customer_ID***: a unique identifier for the customer
- ***Cost***: insurance cost **(target variable)**
- ***Shopping_pt***: number of insurance quotes customers obtained before purchasing insurance **(target variable)**
- ***Group_size***: an ordinal variable that represents the number of people under

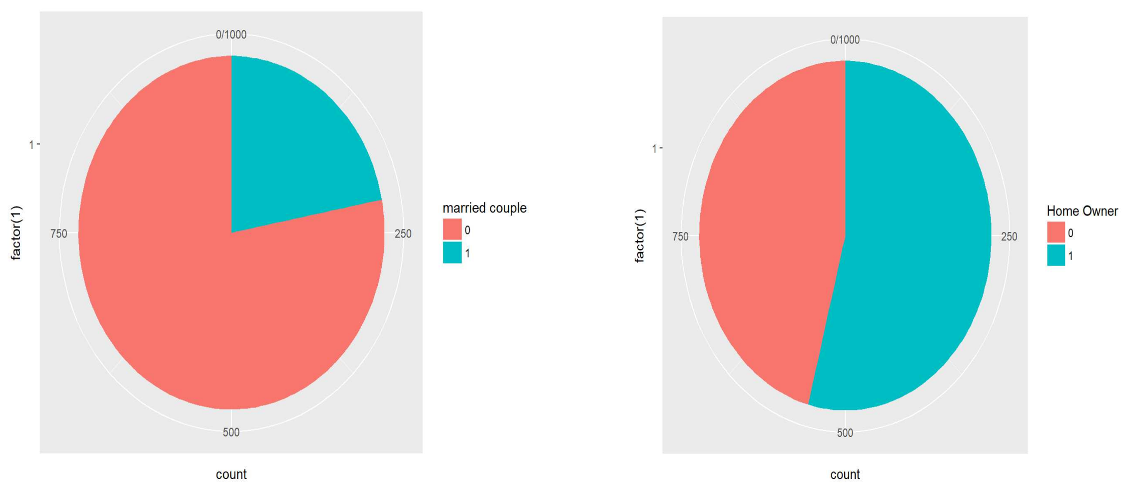
[1] https://www.kaggle.com/c/allstate-purchase-prediction-challenge/data

insurance coverage (can either be 1, 2, 3, 4). **(explanatory variable)**

- ***Homeowner***: a binary variable that represents whether the insured people owns a home (0=”no”, 1= “yes”) **(explanatory variable)**
- ***Car_age***: the age of the car which ranges from (0, 20) **(explanatory variable)**
- ***Married_couple***: a binary variable that represents whether the customer group contains a married couple. **(explanatory variable)**

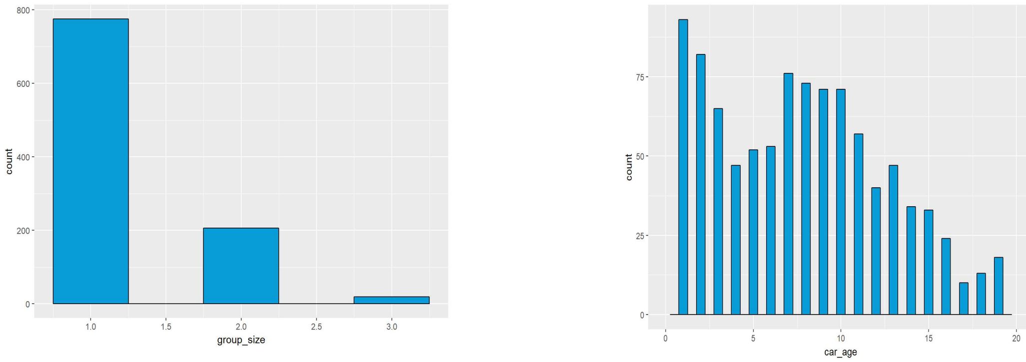
We explored the nature of our dataset by plotting the histograms of four of the explanatory variables and the plots are as follows:

Figure 1.1



The figure above shows a pie chart on the left, which represents the percentage of the customers who are married couples. The pie chart on the right represents the breakdown of the percentage of customers who are homeowners. Less than a quarter of the customers are married, and more than half of the customers owns homes.

Figure 1.2



The figure above shows two histograms for the group size of the insured party on the left and the age of their vehicles on the right. As the group size increases, the count decreases. Furthermore, as the car age increases, the number of counts decreases as well.

Then, we discovered that only 2.3% of the samples have a car age that is greater than 20. We then set the car ages of those sample points whose car age is greater than 20 because most cars' lifespan is less than 20 years. To design our generalized linear regression models, we wanted to see how the explanatory variables correlate with the target variables.

Figure 1.3

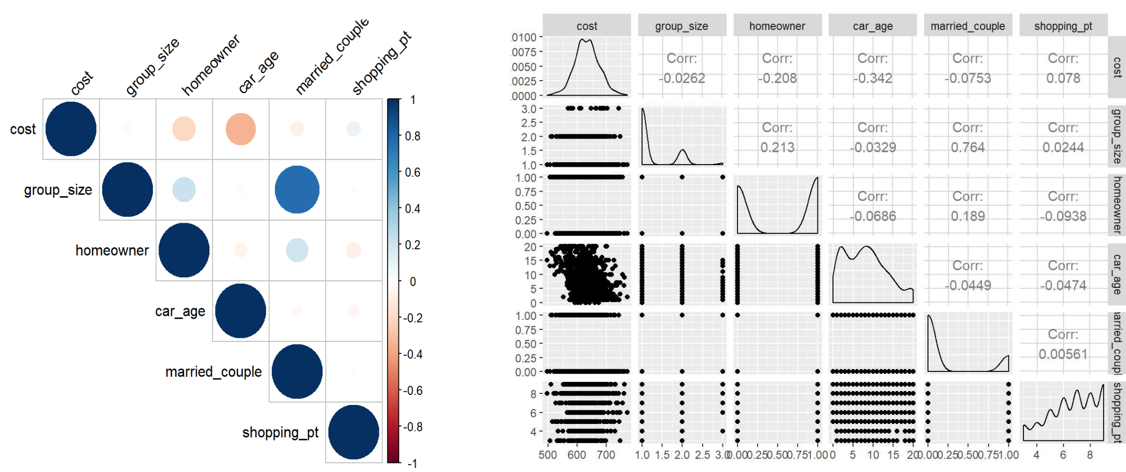


Figure 1.3
The figure above shows a diagram representation of the correlations among the variables on the left and their corresponding numerical values on the right. The intensity of the hue or shade of the colored dot is directly proportional to the correlation

coefficient value. As you can see, the correlation between car_age out of all the other explanatory variables is most closely correlated with the cost with a value of -0.342. Therefore, we took this into account when we built our models. In addition, the distribution of insurance cost seems normally distributed (See Figure 1.4).

Figure 1.4

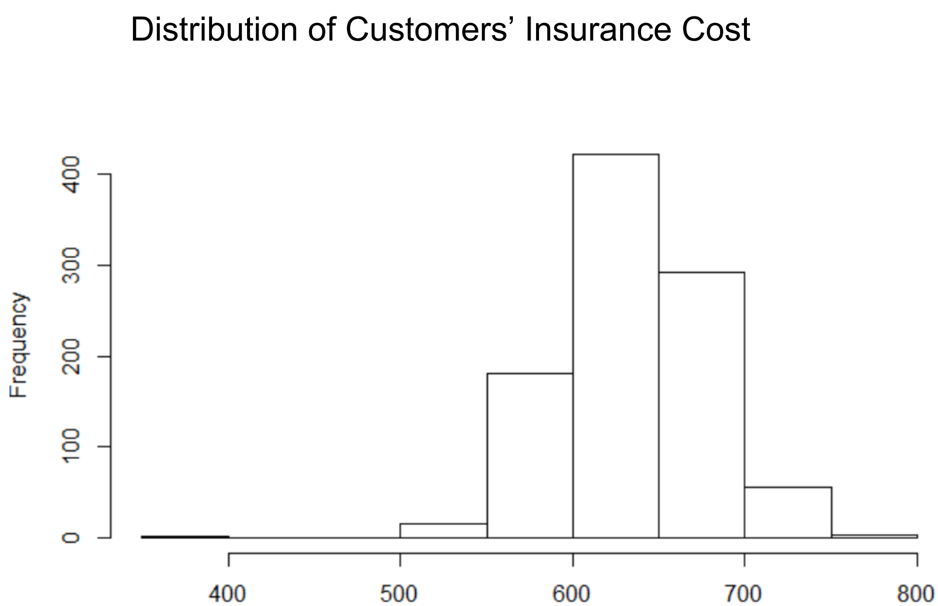


Figure 1.4: The figure above shows the seemingly normally distributed histogram of the customers' insurance cost.

This served as motivation for model building.

Part 3. Model

We built 3 models to analyze the data. The first two models answer our first question: Which of the variables are most explanatory to the target variable cost of insurance? The last model addresses our second question: Does more expensive insurance lead customers obtaining more quotes before purchasing insurance?

Model 1:

The first Bayesian Regression Linear Model hopes to use all relevant variables to predict the customers' cost of insurance:

$$yi|\theta, X \sim indep.N(\beta 1 + \beta 2hi + \beta 3mi + \beta 4ai, \sigma 2)$$

i = 1,..., 1000

$$yi = \beta 1 + \beta 2hi + \beta 3mi + \beta 4ai + \varepsilon i$$

Where:

yi = indicator of cost of insurance for customer i

hi = indicator of homeowner status for customer i

mi = indicator of married status for customer i

ai = indicator of age of car for customer i

The prior is:

$$\beta|X \sim N(0, \sigma_\beta^2 I)$$

$$\sigma^2|X \sim Inv - \chi^2(\nu 0, \sigma 0^2)$$

We then set a flat prior for the Sigma squared and all the Beta, in other words, Let σ_β^2 be large and $\nu 0$ small to make prior nearly noninformative:

$$\beta i \sim indep.N(0, 10000^2)$$

$$\sigma^2 \sim indep.Gamma(0.0001, 0.0001)$$

Model 2:

The second Bayesian Regression Linear Model hopes to use merely car age variables to predict the customers' cost of insurance:

$$yi|\theta, X \sim indep.N(\beta 1 + \beta 2ai, \sigma 2)$$

i = 1,..., 1000

$$yi = \beta 1 + \beta 2ai + \varepsilon i$$

Where:

yi = indicator of cost of insurance for customer i

ai = indicator of age of car for customer i

The prior is:

$$\beta|X \sim N(0, \sigma_{\beta}^{2} I)$$

$$\sigma^2|X \sim Inv - \chi^2(\nu 0, \sigma 0^2)$$

We then set a flat prior for the Sigma squared and all the Beta, in other words, Let σ_{β}^{2} be large and $\nu 0$ small to make prior nearly noninformative:

$$\beta i \sim indep.N(0, 10000^2)$$

$$\sigma^2 \sim indep.Gamma(0.0001, 0.0001)$$

Model 3:

The third Poisson Log Linear Model hopes to use the customers' cost of insurance to predict the customer's shopping point:

$$yi|\beta, Xi \sim indep.Poisson(\lambda i)$$

$$log\lambda i = \beta 1 + Xi\beta 2$$

i = 1,..., 1000

Where:

yi = indicator of shopping point for customer i

xi = indicator of cost of insurance for customer i

The prior is:

$$\beta|X \sim N(0, \sigma_\beta^2 I)$$

We then set a flat prior for the all the Beta, in other words, Let σ_β^2 be large enough to make prior nearly noninformative:

$$\beta i \sim indep. N(0, 100^2)$$

Part 4. Computation and Convergence Test

We used extreme values to initialize 4 chains for model1:

```
d1 <- list( cost = analysis_data$cost
           ,owner = analysis_data$homeowner
           ,married = analysis_data$married_couple
           ,age = analysis_data$car_age
          )

inits1 <- list(
              list(beta_intercept = 1000, beta_homeowner = 1000, beta_married = 1000
                   ,beta_age = 1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

              ,list(beta_intercept = -1000, beta_homeowner = -1000, beta_married = 1000
                   ,beta_age = 1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

              ,list(beta_intercept = 1000, beta_homeowner = 1000, beta_married = -1000
                   ,beta_age = -1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

              ,list(beta_intercept = -1000, beta_homeowner = -1000, beta_married = -1000
                   ,beta_age = -1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)
              )
```

We used extreme value to initialize model2:

```
d2 <- list( cost = analysis_data$cost
           ,age = analysis_data$car_age
          )

inits2 <- list(
              list(beta_intercept = 1000
                   ,beta_age = 1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

              ,list(beta_intercept = -1000
                   ,beta_age = 1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

              ,list(beta_intercept = 1000
                   ,beta_age = -1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

              ,list(beta_intercept = -1000
                   ,beta_age = -1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)
              )
```

We used extreme value to initialize model 3:

```
d3 <- list (num_quotes = analysis_data$shopping_pt
            ,logtime = log(1)
            ,cost_scaled = as.vector(scale(analysis_data$cost, scale=1*sd(analysis_data$cost)))
            )

inits3 <- list(list(beta_intercept = 100 , beta_cost = 100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

               ,list(beta_intercept = -100 , beta_cost = 100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

               ,list(beta_intercept = 100 , beta_cost = -100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

               ,list(beta_intercept = -100 , beta_cost = -100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)

               )

m3 <- jags.model("final_project_point1.bug", d3, inits3, n.chains=4, n.adapt=1000)
```

For each model, we run 1000 iterations for adaptation, 10,000 iterations for burn-in, simulate 20,000 iterations with thin = 10 to get the posterior sample, and 50,000 iterations for DIC. Here is an example:

```
m2 <- jags.model("final_project_cost2.bug",d2,inits2,n.chains=4,n.adapt=1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 1003
##    Total graph size: 3050
##
## Initializing model
```

```
update(m2, 10000)

x2 <- coda.samples(m2, c("beta_intercept","beta_age","sigmasq","cost_rep"),n.iter=20000, thin = 10)

x2_sub <- x2[,c("beta_intercept","beta_age","sigmasq")]
```

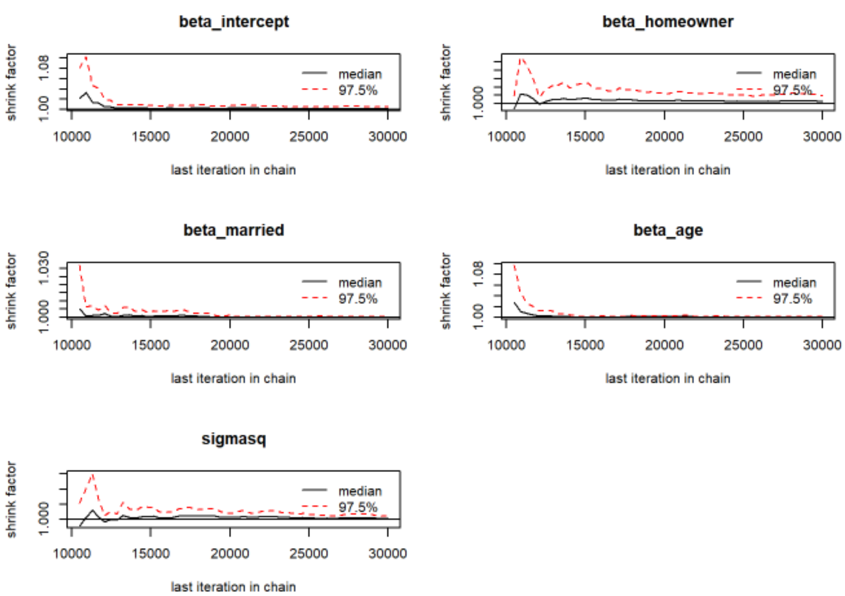
All the model had point estimate less than 1.1 in gelman rubin test and effective size more than 4000, which prove the converge:

Model 1:

```r
gelman.diag(x1_sub,autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##                   Point est. Upper C.I.
## beta_intercept             1       1.01
## beta_homeowner             1       1.00
## beta_married               1       1.00
## beta_age                   1       1.00
## sigmasq                    1       1.00
##
## Multivariate psrf
##
## 1
```

```r
gelman.plot(x1_sub,autoburnin=FALSE)
```




```r
effectiveSize(x1_sub)
```

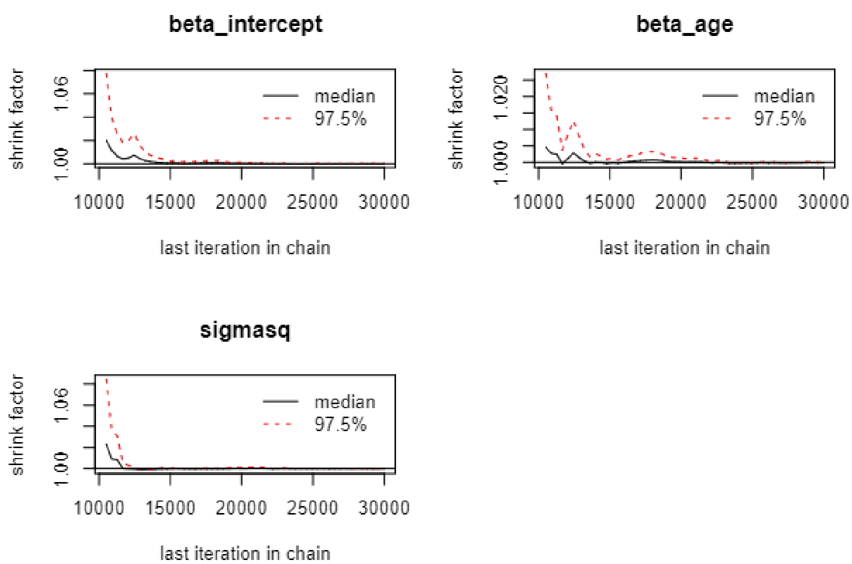
```
## beta_intercept beta_homeowner   beta_married       beta_age        sigmasq
##       6918.692       7711.714       7705.563       7521.025       8000.000
```

Model 2:

```
gelman.diag(x2_sub,autoburnin=FALSE, multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
##                Point est. Upper C.I.
## beta_intercept          1          1
## beta_age                1          1
## sigmasq                 1          1
```

```
gelman.plot(x2_sub,autoburnin=FALSE)
```




```
effectiveSize(x2_sub)
```

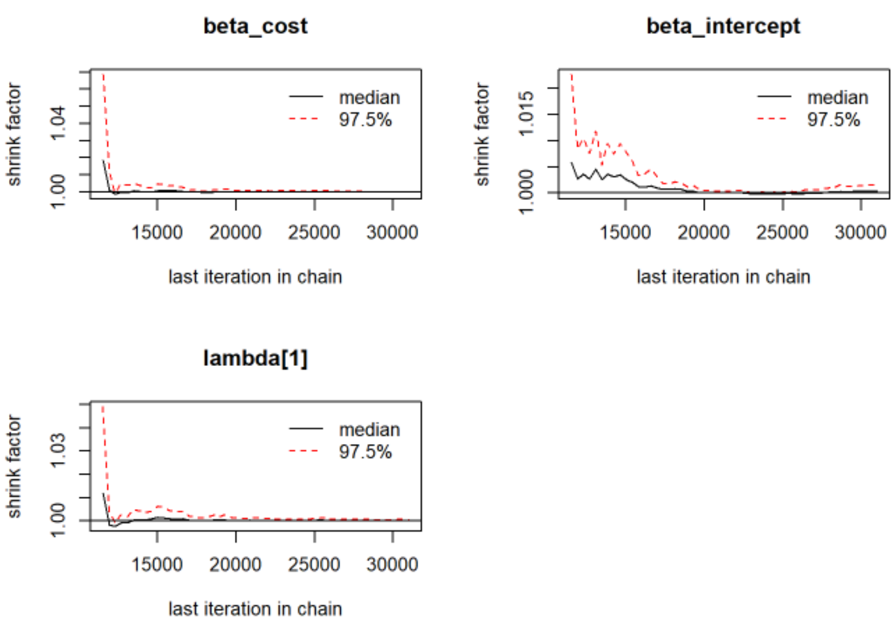
```
## beta_intercept       beta_age        sigmasq
##       7889.800       8185.387       8000.000
```

Model 3:

```
gelman.diag(x3[,1:3], autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##                Point est. Upper C.I.
## beta_cost               1          1
## beta_intercept          1          1
## lambda[1]               1          1
##
## Multivariate psrf
##
## 1
```

```
gelman.plot(x3[,1:3], autoburnin=FALSE)
```




```
effectiveSize(x3[,1:3])
```

```
##      beta_cost beta_intercept      lambda[1]
##       7695.369       8755.014       7761.156
```

Here is the summary of variables in each model.

Model 1:

```
summary(x1_sub)
```

```
##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                    Mean      SD Naive SE Time-series SE
## beta_homeowner   282.02   17.08  0.19092        0.18325
## beta_married     122.04   23.21  0.25948        0.26245
## beta_age          39.51    1.24  0.01386        0.01387
## sigmasq        90960.88 4087.85 45.70358       45.78405

## 2. Quantiles for each variable:
##
##                    2.5%      25%      50%      75%    97.5%
## beta_homeowner   247.76   270.42   282.05   293.50   315.67
## beta_married      76.08   106.69   122.20   137.65   167.38
## beta_age          37.04    38.66    39.51    40.35    41.91
## sigmasq        83206.25 88191.93 90892.40 93620.13 99331.61
```

Model 2:

```
summary(x2_sub)
```

```
##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                    Mean      SD Naive SE Time-series SE
## beta_intercept  655.898  2.2851 0.025548       0.025732
## beta_age         -2.749  0.2376 0.002656       0.002627
## sigmasq        1552.360 69.2025 0.773708       0.773834
##
## 2. Quantiles for each variable:
##
##                    2.5%      25%     50%      75%    97.5%
## beta_intercept  651.400  654.384  655.91  657.396  660.456
## beta_age         -3.223   -2.907   -2.75   -2.591   -2.273
## sigmasq        1424.534 1504.221 1549.57 1597.922 1693.777
```

Model 3:

```
summary(x3[,1:2])
```

```
##
## Iterations = 11010:31000
## Thinning interval = 10
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                   Mean      SD  Naive SE Time-series SE
## beta_cost      0.02051 0.01221 0.0001365      0.0001394
## beta_intercept 1.91241 0.01225 0.0001370      0.0001316
## 2. Quantiles for each variable:
##
##                     2.5%     25%     50%     75%   97.5%
## beta_cost      -0.003667 0.01228 0.02048 0.02881 0.04439
## beta_intercept  1.888226 1.90432 1.91244 1.92056 1.93647
```

Parts 5 & 6. Model Assessment and Result

As we mentioned before, the first two models answer the question 1 - which of the variables are most explainable to the target variable cost of insurance. Transforming the question 1 into statistic language, we want to ask: 1. Are those betas of each explainable variable equal to 0 in each model? 2. Different model chose different variables to explain the target, which model had lowest DIC? We used 95% posterior confidence interval to answer the former, and DIC to answer the latter.

For model 1, we can find the beta of homeowner, car age 95% confidence interval do not include the 0, but the beta of married status 95% confidence interval do include 0:

```
post.samp1 <- as.matrix(x1)

##The 95% confidence interval of beta_homeowner does not include 0
#The mean
mean(post.samp1[,"beta_homeowner"])
```

```
## [1] -18.76625
```

```
#The 95% confidence interval
quantile(post.samp1[,"beta_homeowner"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -23.61595 -13.86279
```

```
##The 95% confidence interval of beta_married does not include 0
#The mean
mean(post.samp1[,"beta_married"])
```

```
## [1] -5.036647
```

```
#The 95% confidence interval
quantile(post.samp1[,"beta_married"], c(0.025,0.975))
```

```
##       2.5%      97.5%
## -10.763138   0.830629
```

```r
##The 95% confidence interval of beta_age does not include 0
#The mean
mean(post.samp1[,"beta_age"])
```

```
## [1] -2.890251
```

```r
#The 95% confidence interval
quantile(post.samp1[,"beta_age"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -3.353562 -2.432684
```

For model 2, we can find the beta of car age 95% confidence interval does not include the 0:

```r
##The 95% confidence interval of beta_age does not include 0
#The mean
mean(post.samp2[,"beta_age"])
```

```
## [1] -2.748917
```

```r
#The 95% confidence interval
quantile(post.samp2[,"beta_age"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -3.223461 -2.272747
```

In conclusion, car age and homeowner are explanatory to insurance cost, while married status is not.

Next, we want to find which model had lower DIC:

```
dic.samples(m1,50000)
```

```
## Mean deviance:  10121
## penalty 4.984
## Penalized deviance: 10125
```

```
dic.samples(m2,50000)
```

```
## Mean deviance:  10184
## penalty 3.016
## Penalized deviance: 10187
```

We can find model 1 has lower DIC, but the difference is not too much. car age is probably a more dominant factor than homeownership status and marital status, because the latter two factors only improve the penalized deviance by ~0.6%.

Model 3 was built to answer question 2 - more expensive insurance is, the more number of quotes customers obtain before purchasing insurance. Transforming into statistic language, we want to know: 1. Does the beta of car age equal to 1? 2. Does the model have underdispersion issue?

After checking the 95% posterior confidence interval of beta-cost, we can find it include the 1, which indicate it is not statistically significant:

```r
post.samp3 <- as.matrix(x3)

##The 95% confidence interval of beta_cost does not include 1
quantile(exp(post.samp3[,"beta_intercept"]),c(0.025,0.975))
```

```
##     2.5%    97.5%
## 6.607635 6.934211
```

```r
##The 95% confidence interval of beta_cost includes 1
quantile(exp(post.samp3[,"beta_cost"]),c(0.025,0.975))
```

```
##      2.5%     97.5%
## 0.9963401 1.0453881
```

We also checked the Chi-square discrepancy for model 3, the posterior predictive p-value is close to 1, which indicate overdispersion problem:

```r
post.samp3 <- as.matrix(x3)

lambdas <- post.samp3[,paste("lambda[",1:nrow(analysis_data),"]", sep="")]

num_quotes_srep <- post.samp3[,paste("num_quotes_rep[",1:nrow(analysis_data),"]", sep="")]

Tchi <- numeric(nrow(num_quotes_srep))
Tchirep <- numeric(nrow(num_quotes_srep))

for(s in 1:nrow(num_quotes_srep)) {
  Tchi[s] <- sum((analysis_data$shopping_pt - lambdas[s,])^2/lambdas[s,])
  Tchirep[s] <- sum((num_quotes_srep[s,]-lambdas[s,])^2/lambdas[s,])
}

mean(Tchirep >= Tchi)
```

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

In conclusion for question 2, given beta_cost's 95% interval and Chi-Square test, the number of quotes customers obtained does not depend on the cost of insurance.

Part 7. Contribution

All team members contributed substantially to this project. Throughout the entire project, we each performed approximately the same amount work. All three of us were actively involved in brainstorming the proposal, performing mathematical analyses, recording the presentation, and writing the report.

Part 8. Reference

Link to Data: https://www.kaggle.com/c/allstate-purchase-prediction-challenge

Latex Function: https://www.codecogs.com/latex/eqneditor.php

Part 9.

Appendices

STAT 578 - Final Project

Part 1 - Load in the data and libraries

```
library("rjags")
```

```
## Loading required package: coda
## Linked to JAGS 4.3.0
## Loaded modules: basemod,bugs
```

```
library("lattice")

setwd("Z:/MSC-DS/2017 - Fall/STAT 578 - Advanced Bayesian Modeling/Final Project")

#load data, initialize starting values and set up a model
project_data <- read.csv(file="stat_578_data.csv")
```

Part 2(a)

- Take a subset of the data. Specifically, 1,000 random sample customers

```
#Randomly select 1,000 rows of data
set.seed(123)
analysis_data = project_data[sample(nrow(project_data), 1000),]
```

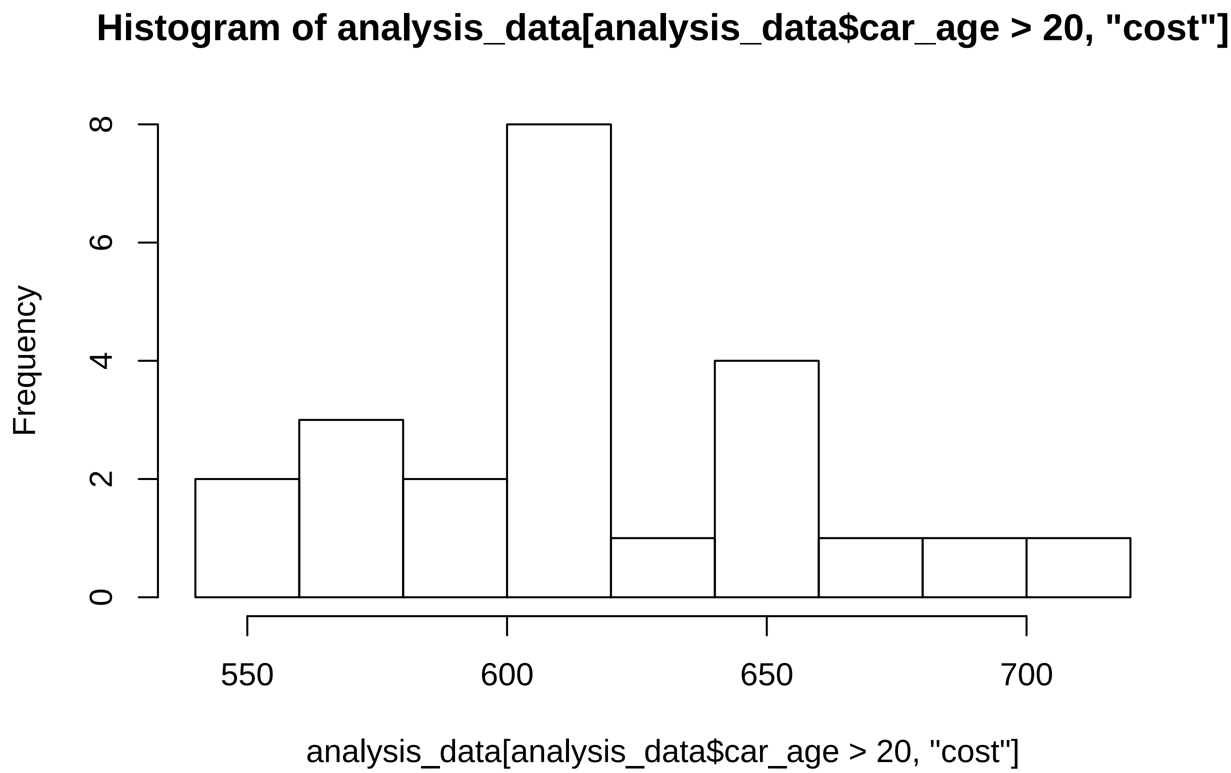
Part 2(b)

- For car_age > 20, change it to 20

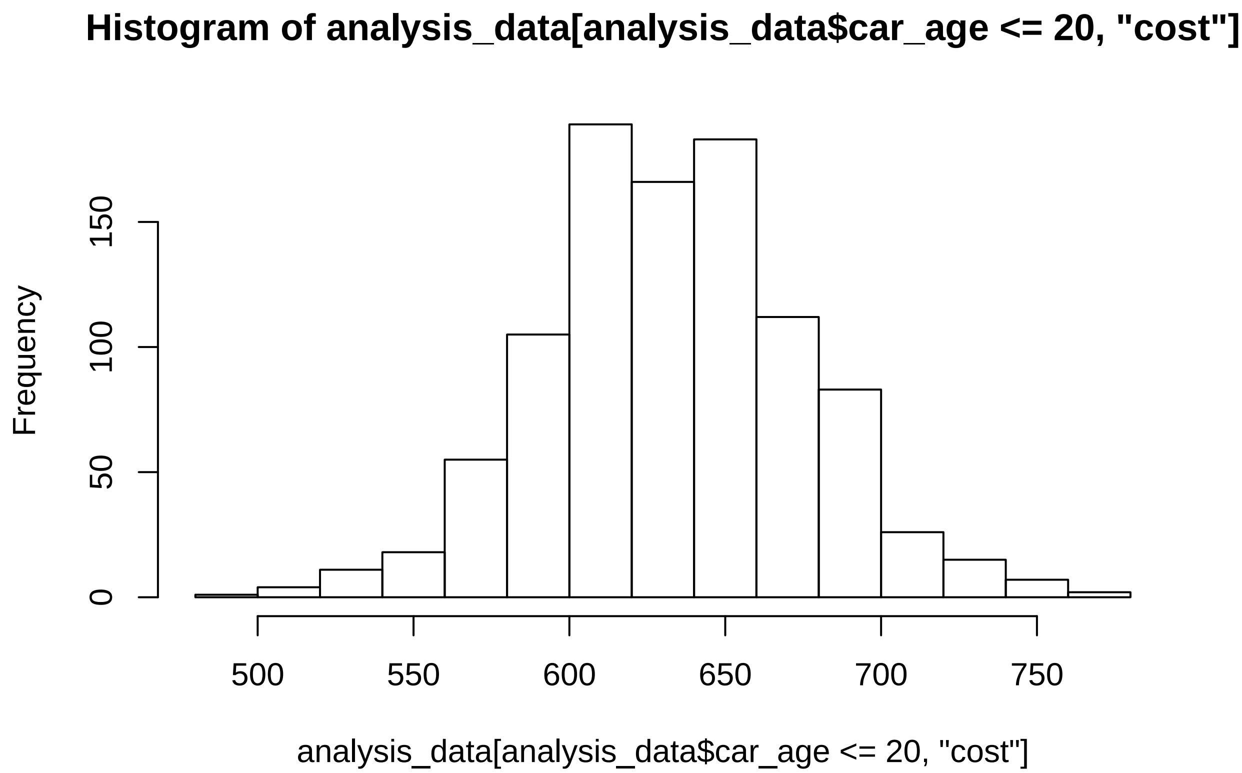
```
#For car_age > 20, set it to 20 for the three reasons below:
#(1) most cars' useful life is less than 20 years.
#(2) the number of vehicles older than 20 make up about 2.3 % of the overall data
length(analysis_data[analysis_data$car_age > 20, "car_age"])/1000
```

```
## [1] 0.023
```

```
#(3) the insurance cost for car_age > 20 seems to have the same mean as car_age <= 20
hist(analysis_data[analysis_data$car_age > 20, "cost"])
```




```r
hist(analysis_data[analysis_data$car_age <= 20, "cost"])
```




```
#So, for car_age > 20, set it to 20.
index <- analysis_data$car_age > 20
analysis_data$car_age[analysis_data$car_age>20] <- 20
```

Part 3(a)

-Fit a model for y[i] ~ beta_intercept + beta_homeowner*owner[i] + beta_married_couple[i] + beta_age x car_age[i] -Check for convergence of the regression coefficients

```
d1 <- list( cost = analysis_data$cost
            ,owner = analysis_data$homeowner
            ,married = analysis_data$married_couple
            ,age = analysis_data$car_age
           )

inits1 <- list(
              list(beta_intercept = 1000, beta_homeowner = 1000, beta_married = 1000
                   ,beta_age = 1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

              ,list(beta_intercept = -1000, beta_homeowner = -1000, beta_married = 1000
                   ,beta_age = 1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

              ,list(beta_intercept = 1000, beta_homeowner = 1000, beta_married = -1000
                   ,beta_age = -1000, sigmasqinv = 1000000,
```

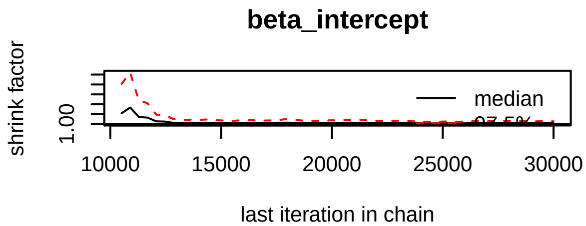
```r
                    .RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

               ,list(beta_intercept = -1000, beta_homeowner = -1000, beta_married = -1000
                    ,beta_age = -1000, sigmasqinv = 0.0000001,
                    .RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)
               )


m1 <- jags.model("final_project_cost1_xly_updated_v1.bug",d1,inits1,n.chains=4,n.adapt=1000)
```

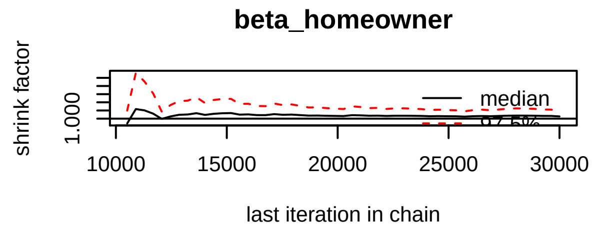
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 1005
##    Total graph size: 5116
##
## Initializing model
```

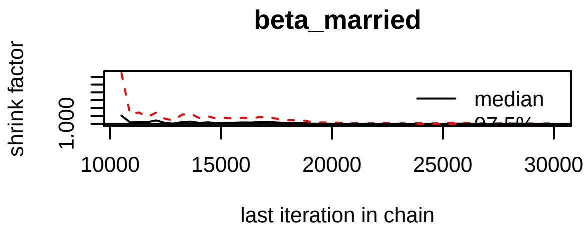
```r
update(m1, 10000)

x1 <- coda.samples(m1, c("beta_intercept", "beta_homeowner","beta_married","beta_age","sigmasq","cost_r

x1_sub <- x1[,c("beta_intercept", "beta_homeowner","beta_married","beta_age","sigmasq")]

gelman.diag(x1_sub,autoburnin=FALSE)
```

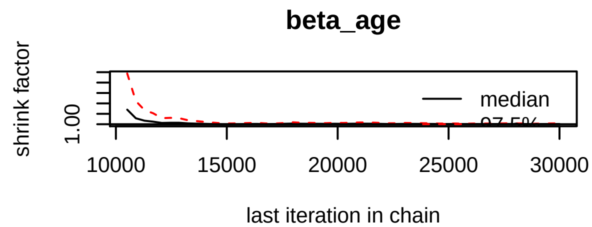
```
## Potential scale reduction factors:
##
##                Point est. Upper C.I.
## beta_intercept          1       1.01
## beta_homeowner          1       1.00
## beta_married            1       1.00
## beta_age                1       1.00
## sigmasq                 1       1.00
##
## Multivariate psrf
##
## 1
```

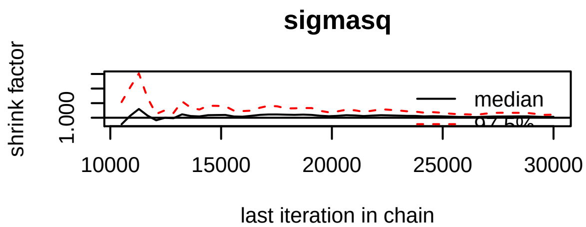
```r
gelman.plot(x1_sub,autoburnin=FALSE)
```












```r
effectiveSize(x1_sub)
```

```
## beta_intercept beta_homeowner   beta_married       beta_age        sigmasq
##       6918.692       7711.714       7705.563       7521.025       8000.000
```

Part 3(b)

- Show summary of beta_homeowner, beta_married, beta_age, and sigmasq

```r
summary(x1_sub)
```

```
##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                    Mean      SD Naive SE Time-series SE
## beta_intercept  668.268  2.7155 0.030361        0.03262
## beta_homeowner  -18.766  2.5101 0.028063        0.02858
## beta_married     -5.037  2.9406 0.032877        0.03360
## beta_age         -2.890  0.2322 0.002596        0.00270
## sigmasq        1457.853 64.6243 0.722522        0.72250
```

```
##
## 2. Quantiles for each variable:
##
##                      2.5%      25%      50%      75%     97.5%
## beta_intercept  662.977  666.410  668.251  670.140  673.5288
## beta_homeowner  -23.616  -20.497  -18.781  -17.086  -13.8628
## beta_married    -10.763   -7.021   -5.081   -3.067    0.8306
## beta_age         -3.354   -3.047   -2.889   -2.730   -2.4327
## sigmasq        1335.746 1413.449 1456.562 1500.159 1588.5157
```

Part 3(c)

- Check 95% confidence interval for statistical significance for beta_homeowner, beta_married and beta_age

```r
post.samp1 <- as.matrix(x1)

##The 95% confidence interval of beta_homeowner does not include 0
#The mean
mean(post.samp1[,"beta_homeowner"])
```

```
## [1] -18.76625
```

```r
#The 95% confidence interval
quantile(post.samp1[,"beta_homeowner"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -23.61595 -13.86279
```

```r
##The 95% confidence interval of beta_married does not include 0
#The mean
mean(post.samp1[,"beta_married"])
```

```
## [1] -5.036647
```

```r
#The 95% confidence interval
quantile(post.samp1[,"beta_married"], c(0.025,0.975))
```

```
##       2.5%      97.5%
## -10.763138   0.830629
```

```r
##The 95% confidence interval of beta_age does not include 0
#The mean
mean(post.samp1[,"beta_age"])
```

```
## [1] -2.890251
```

```r
#The 95% confidence interval
quantile(post.samp1[,"beta_age"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -3.353562 -2.432684
```

Part 3(d)

- Check dic for model in Part 3(c)

```r
dic.samples(m1,50000)
```

```
## Mean deviance:  10121
## penalty 4.984
## Penalized deviance: 10125
```

Part 4(a)

-Fit a model for y[i] ~ beta_intercept + beta.age x car_age[i]
-Check for convergence of the regression coefficients

```r
#Coda summary of my results for the monitored parameters


d2 <- list( cost = analysis_data$cost
            ,age = analysis_data$car_age
           )

inits2 <- list(
              list(beta_intercept = 1000
                   ,beta_age = 1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

              ,list(beta_intercept = -1000
                   ,beta_age = 1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

              ,list(beta_intercept = 1000
                   ,beta_age = -1000, sigmasqinv = 1000000,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

              ,list(beta_intercept = -1000
                   ,beta_age = -1000, sigmasqinv = 0.0000001,
                   .RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)
              )


m2 <- jags.model("final_project_cost2.bug",d2,inits2,n.chains=4,n.adapt=1000)
```

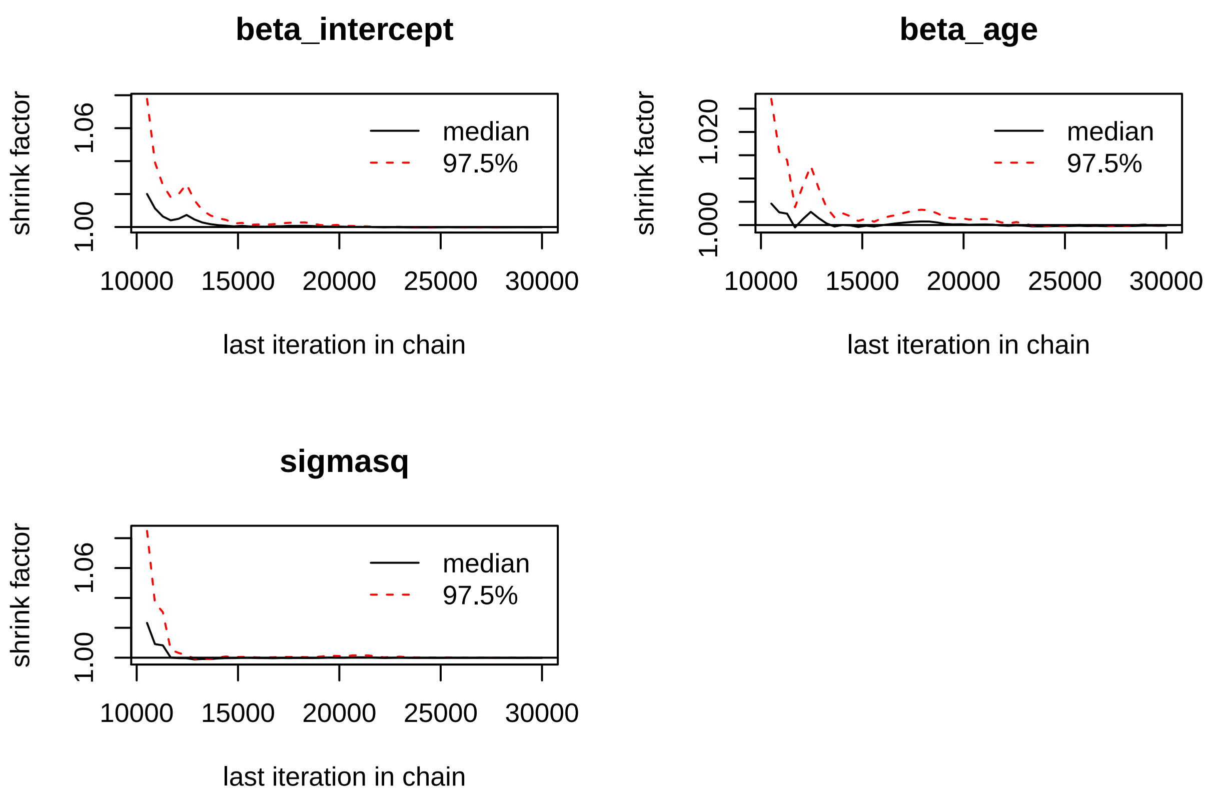
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 1003
##    Total graph size: 3050
##
## Initializing model
```

```r
update(m2, 10000)

x2 <- coda.samples(m2, c("beta_intercept","beta_age","sigmasq","cost_rep"),n.iter=20000, thin = 10)

x2_sub <- x2[,c("beta_intercept","beta_age","sigmasq")]
```

```r
gelman.diag(x2_sub,autoburnin=FALSE, multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
##                Point est. Upper C.I.
## beta_intercept          1          1
## beta_age                1          1
## sigmasq                 1          1
```

```r
gelman.plot(x2_sub,autoburnin=FALSE)
```




```r
effectiveSize(x2_sub)
```

```
## beta_intercept       beta_age        sigmasq
##       7889.800       8185.387       8000.000
```

Part 4(b)

- Show summary of beta_intercept, beta_age, and sigmasq

```r
summary(x2_sub)
```

```
##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 4
```

```
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                     Mean      SD Naive SE Time-series SE
## beta_intercept  655.898  2.2851 0.025548       0.025732
## beta_age         -2.749  0.2376 0.002656       0.002627
## sigmasq        1552.360 69.2025 0.773708       0.773834
##
## 2. Quantiles for each variable:
##
##                     2.5%      25%     50%      75%    97.5%
## beta_intercept  651.400  654.384  655.91  657.396  660.456
## beta_age         -3.223   -2.907   -2.75   -2.591   -2.273
## sigmasq        1424.534 1504.221 1549.57 1597.922 1693.777
```

Part 4(c)

- Check 95% confidence interval for statistical significance for beta_intercept and beta_age

```r
post.samp2 <- as.matrix(x2)

##The 95% confidence interval of beta_intercept does not include 0
#The mean
mean(post.samp2[,"beta_intercept"])
```

```
## [1] 655.8977
```

```r
#The 95% confidence interval
quantile(post.samp2[,"beta_intercept"], c(0.025,0.975))
```

```
##     2.5%    97.5%
## 651.3997 660.4561
```

```r
##The 95% confidence interval of beta_age does not include 0
#The mean
mean(post.samp2[,"beta_age"])
```

```
## [1] -2.748917
```

```r
#The 95% confidence interval
quantile(post.samp2[,"beta_age"], c(0.025,0.975))
```

```
##      2.5%     97.5%
## -3.223461 -2.272747
```

Part 4(d)

- Check dic for model in Part 4(c)

```r
dic.samples(m2,50000)
```

```
## Mean deviance:  10184
## penalty 3.016
## Penalized deviance: 10187
```

Part 5(a)

-Fit the following loglinear model
num_quotes[i] ~ dpois(lambda[i])
log(lambda[i]) <- logtime + beta_intercept + beta_cost*cost_scaled[i]
-Check for convergence for statistical significance for the regression coefficients

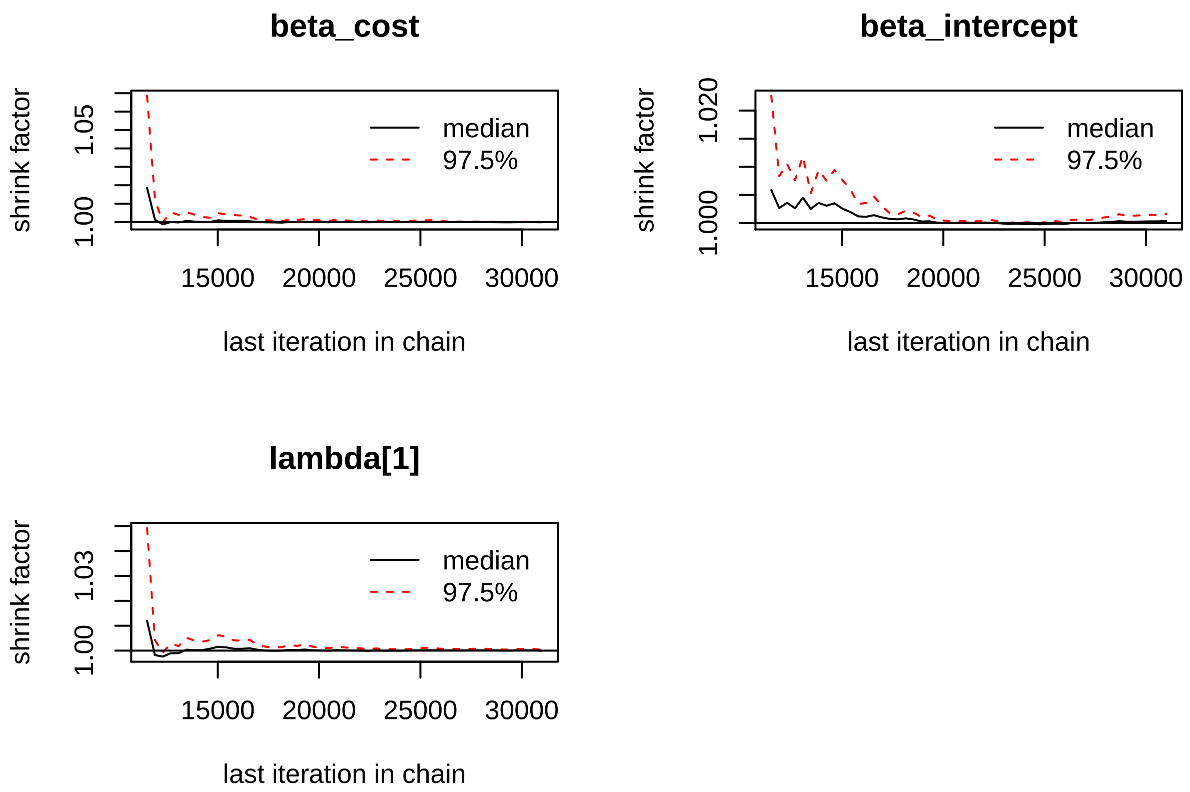
```
d3 <- list (num_quotes = analysis_data$shopping_pt
            ,logtime = log(1)
            ,cost_scaled = as.vector(scale(analysis_data$cost, scale=1*sd(analysis_data$cost)))
            )

inits3 <- list(list(beta_intercept = 100 , beta_cost = 100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 101)

               ,list(beta_intercept = -100 , beta_cost = 100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 103)

               ,list(beta_intercept = 100 , beta_cost = -100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 105)

               ,list(beta_intercept = -100 , beta_cost = -100
               ,.RNG.name = "base::Mersenne-Twister", .RNG.seed = 107)

              )

m3 <- jags.model("final_project_point1.bug", d3, inits3, n.chains=4, n.adapt=1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1000
##    Unobserved stochastic nodes: 1002
##    Total graph size: 3614
##
## Initializing model
```

```
update(m3, 10000)

x3 <- coda.samples(m3, c("beta_intercept","beta_cost", "num_quotes_rep", "lambda"), n.iter=20000, thin =

gelman.diag(x3[,1:3], autoburnin=FALSE)
```

```
## Potential scale reduction factors:
##
##                Point est. Upper C.I.
## beta_cost               1          1
## beta_intercept          1          1
## lambda[1]               1          1
##
## Multivariate psrf
##
## 1
```

```r
gelman.plot(x3[,1:3], autoburnin=FALSE)
```




```r
effectiveSize(x3[,1:3])
```

```
##      beta_cost beta_intercept      lambda[1]
##       7695.369       8755.014       7761.156
```

Part (5)(b)

- Show summary of beta_cost and beta_intercept

```r
summary(x3[,1:2])
```

```
##
## Iterations = 11010:31000
## Thinning interval = 10
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                   Mean      SD  Naive SE Time-series SE
## beta_cost      0.02051 0.01221 0.0001365      0.0001394
## beta_intercept 1.91241 0.01225 0.0001370      0.0001316
##
```

```
## 2. Quantiles for each variable:
##
##                     2.5%     25%     50%     75%   97.5%
## beta_cost      -0.003667 0.01228 0.02048 0.02881 0.04439
## beta_intercept  1.888226 1.90432 1.91244 1.92056 1.93647
```

Part (5)(c)

- Check 95% confidence interval for statistical significance for beta_intercept and beta_cost

```r
post.samp3 <- as.matrix(x3)

##The 95% confidence interval of beta_cost does not include 1
quantile(exp(post.samp3[,"beta_intercept"]),c(0.025,0.975))
```

```
##     2.5%    97.5%
## 6.607635 6.934211
```

```r
##The 95% confidence interval of beta_cost includes 1
quantile(exp(post.samp3[,"beta_cost"]),c(0.025,0.975))
```

```
##      2.5%     97.5%
## 0.9963401 1.0453881
```

Part (5)(d)

-The p-value for the Chi-square test is 1. This indicates that the variance of the data is smaller than what the Poisson distribution assumes.

```r
post.samp3 <- as.matrix(x3)

lambdas <- post.samp3[,paste("lambda[",1:nrow(analysis_data),"]", sep="")]

num_quotes_srep <- post.samp3[,paste("num_quotes_rep[",1:nrow(analysis_data),"]", sep="")]

Tchi <- numeric(nrow(num_quotes_srep))
Tchirep <- numeric(nrow(num_quotes_srep))

for(s in 1:nrow(num_quotes_srep)) {
  Tchi[s] <- sum((analysis_data$shopping_pt - lambdas[s,])^2/lambdas[s,])
  Tchirep[s] <- sum((num_quotes_srep[s,]-lambdas[s,])^2/lambdas[s,])
}

mean(Tchirep >= Tchi)
```

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

Part 5(e)

- Check dic for model in Part 5

```r
dic.samples(m3,50000)
```

```
## Mean deviance:  4244
## penalty 2.003
## Penalized deviance: 4246
```

```sql
--STEP 1
--Inport train.csv as a table to the Microsoft SQL Server and name the table train.

--STEP 2
--From the train table, for each customer extract the row with the maximum shopping_pt.
--The shopping_pt column is the number of quotes a custombber obtained, where 1 means the
--first quote and 12 means the twelfth quote. In the provided data, customers purchased a policy
--in the last quote

SELECT table_A.[customer_ID]
      , table_A.[shopping_pt]
      , table_A.[group_size]
      , table_A.[homeowner]
      , table_A.[car_age]
      , table_A.[married_couple]
      , table_A.[cost]
      , table_A.[duration_previous]
INTO stat_578_data
FROM train table_A inner join
(SELECT [customer_ID]
,MAX([shopping_pt]) AS max_pt
FROM train
GROUP BY [customer_ID]) table_B
on table_A.customer_ID=table_B.customer_ID and table_A.shopping_pt=table_B.max_pt
--(97009 rows affected)
```