MATH2269 - Final Project

Implementing Multiple Linear Regression model to Predict

the Crime Index for a County

using Demographic and Economic information

Contents

1. Introduction

Since the beginning of humanity, we have been living in groups. These groups eventually grew into what we today call as societies. Even with diversities among the society all across the world, few things are always common. Crimes committed by few of the members in the society play a negative but impactful role in well-being of everyone in that society.

In spite of the law enforcements deployed, crime still remains a big threat. We believe, if the frequency of occurrence of the crime can be explained, it would be step ahead to control and reduce the crimes. If the policy making people in the government have knowledge about which factors of a society lead to crime and how significant are these factors, they would be in much better position to tackle this social issue of crime.

For this reason, in this analysis we have tried to model the crime index using multiple demographic features like age, savings of people, population density, proportion of people under the poverty line, etc. So, the problem statement for this analysis will be given as below:

1.1. Problem Statement

"How can crime index for a particular county be explained using the demographic and economic information for that county."

1.2. Overview

In this report we will be implementing a Bayesian Multiple Linear Regression Model to predict the crime index for a county using the demographic and economic data. While implementing this model, we will take into consideration the data available for presidential elections of 1992 in the United States and also our own belief about how various demographic and economic features can affect the crime index in the society.

1.3. Data

The data used for this analysis has been sourced from the US Census Bureau. This data was originally meant to show the percentage of people who have voted for Bill Clinton from every county in US during the presidential elections of 1992. For this analysis, to model the crime index, subset(relevant columns) of this data will be used. Columns describing information relevant to the elections like county name, percent of people voting for Bill Clinton and count of nursing homes in the county have not been considered.

The data used for this analysis consists of 7 variables. Out of these 7 variables, *crime_index* is the dependent variable and *median_age*, *savings*, *per_capita_income*, *poverty_prcnt*, *veterans_prcnt* and *population_density* are the independent variables(predictors). The size of data available is of 2,702 records.

1.4. Multiple Linear Regression Equation

Considering the above mentioned dependent and independent variables, the linear regression equation can be given as:

$$\textbf{crime\_index} = \beta_0 + \beta_1 * \textbf{median\_age} + \beta_2 * \textbf{savings} + \beta_3 * \textbf{per\_capita\_income} + \beta_4 * \textbf{poverty\_prcnt} + \beta_5 * \textbf{veterans\_prcnt} + \beta_6 * \textbf{population\_density}$$

In the above equation, β_0 is the intercept and β_1, β_2, β_3, β_4, β_5, β_6 are the coefficients for predictors *median_age*, *savings*, *per_capita_income*, *poverty_prcnt, veterans_prcnt* and *population_density* respectively.

1.5. Methodology

In this analysis, we will try to find the Bayesian estimate for coefficient of every parameter (predictor). The estimation will be done using MCMC (Markov Chain Monte Carlo) simulation which will be implemented using JAGS (Just Another Gibbs Sampler). While estimating the coefficients, we will include our belief as the prior information for each parameter along with the data available. We will then check the MCMC diagnostics for each parameter to ensure that MCMC simulation has been correctly performed. This is involve checking the MCMC chains for their representativeness, accuracy, and efficiency. Then the Bayesian point estimate would be found for every coefficient by observing the respective posterior distributions. After obtaining these Bayesian estimates, we will use the above regression equation to predict the crime index using different values for *median_age*, *savings*, *per_capita_income*, *poverty_prcnt, veterans_prcnt* and *population_density*. Lastly, using the obtained regression model and available data, predicted crime index will be computed. To assess the accuracy and effectiveness of the model, the KDE(Kernel Density Estimate) of observed crime index and the predicted crime index will be compared.

We will start the analysis by performing a descriptive check on data available.

2. Descriptive Check

2.1. Distribution Check

To define the Bayesian model for above specified regression equation, it is important to know the distribution of the variable which we are trying to predict, which in this case would be *crime_index*. So, the KDE for crime index can be given as below:

(P.T.O.)

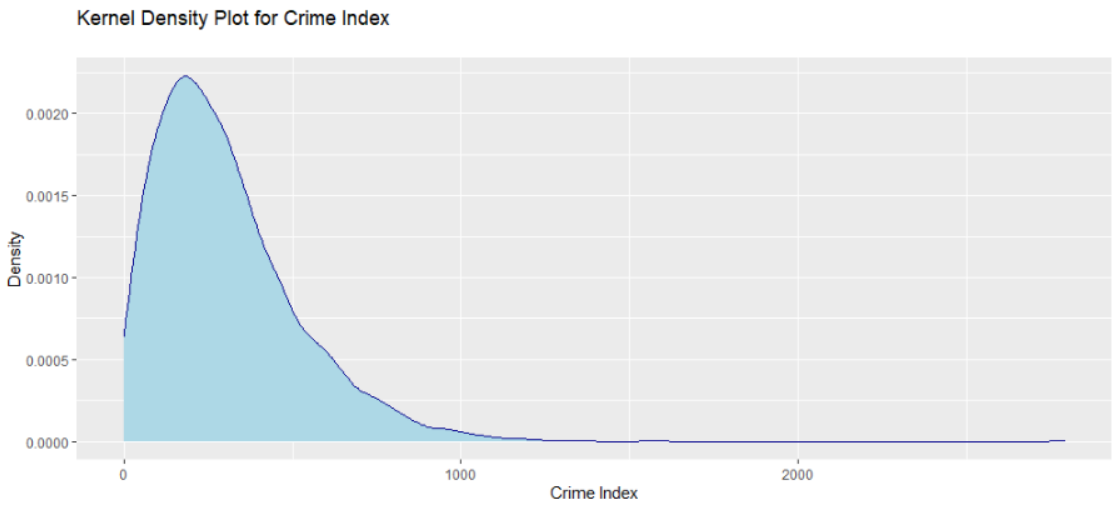


Figure 1: Kernel Density Estimate plot for *crime_index*

From the above output, it can be seen that the curve corresponds to a gamma distribution. So, for this analysis, it would be possible to consider a normal-gamma model where the posterior distribution and likelihood will follow a gamma distribution and the prior distributions can be specified as in the form of normal distributions.

Next, for a regression model, it is assumed that the independent variables(predictors) follow a gaussian distribution. To check if this assumption is true for the predictors of this analysis, the KDE for all the predictors is as below:

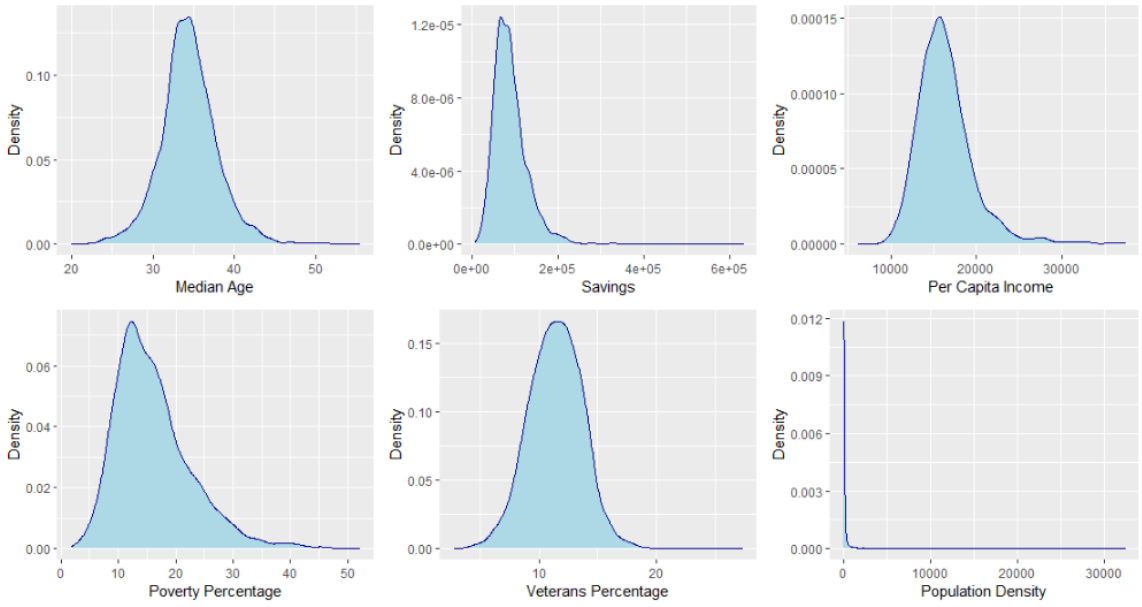


Figure 2: Kernel Density Estimate plot for independent variables *median_age*, *savings*, *per_capita_income*, *poverty_prcnt*, *veterans_prcnt* and *population_density*

From the output, it can be seen that all the predictors are more or less right skewed. The distributions for *median_age*, *per_capita_income*, *poverty_prcnt* and *veterans_prcnt* are slightly right skewed, whereas the distributions for *savings* and *population_density* are heavily right skewed.

We suspect that behavior of the distribution of predictors can affect the results given by this model.

2.2. Outliers Check

Now, using scatter plots, we will check the relation of dependent variable *crime_index* with every independent variable. The respective scatter plots are as follows:

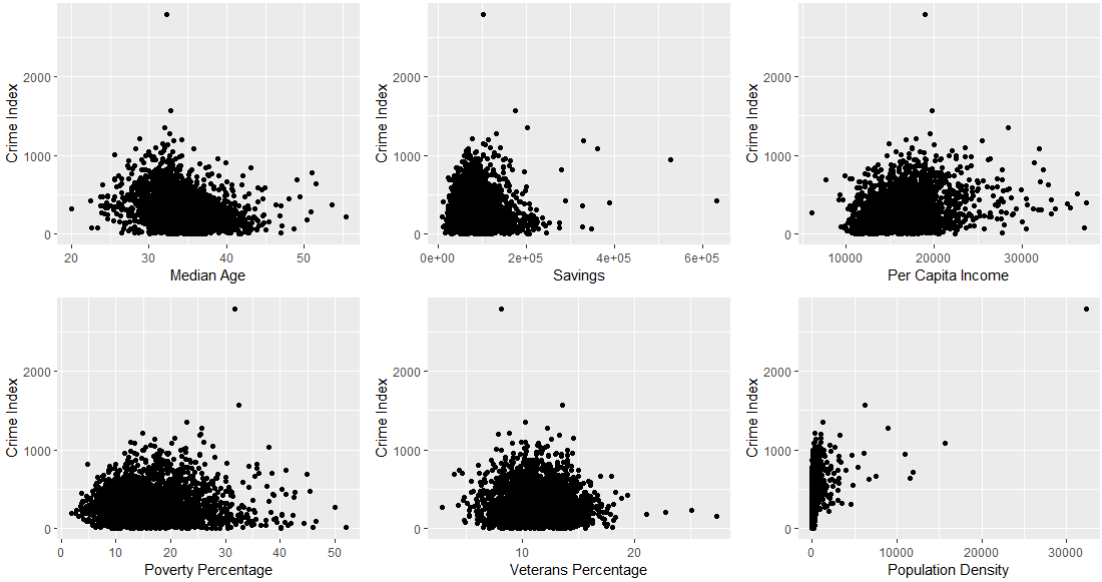


Figure 3: Scatter plots to represent the relation of dependent variable with independent variables

From the above output, it can be seen that there is no significant relation of dependent variable *crime_index* with any of the independent variable. This means that, in the regression model, no independent variable will dominate the behavior of dependent variable *crime_index*. From the output, it can also be seen that data points are concentrated within a certain range. This will restrict the model from capturing the behavior of values which are not present in the data.

Lastly, we can see presence of multiple outlying values. The presence of these outlying values can be confirmed by plotting a box plot. The box plot for all the variables can be given as below:

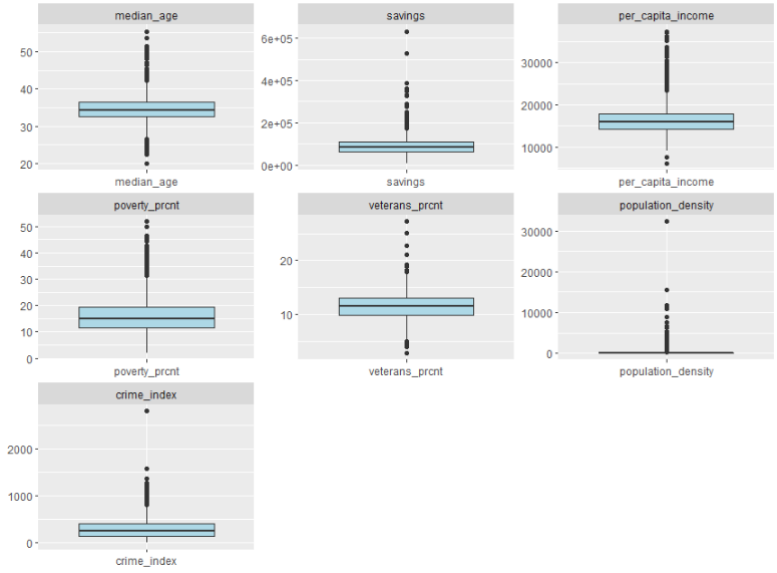


Figure 4: Box plots of all the variables

From the output, it can be seen that multiple outlying values are present for all the variables. As the regression model searches for an optimal line by minimizing the errors computed for each data point, the outlying values can negatively the affect this process and the obtained optimal line might not be as accurate.

Here, as the data consists of just 2,702 records, we will not eliminate the outlying values.

2.3. Correlation Check

For a regression problem, it assumed that there is no correlation between the independent variables. This is because, by including 2 highly correlated variables, the model receives redundant information. This can cause the correlated variables to dominate the behavior of dependent variable. So, the correlation between independent variables for this model can be given as:

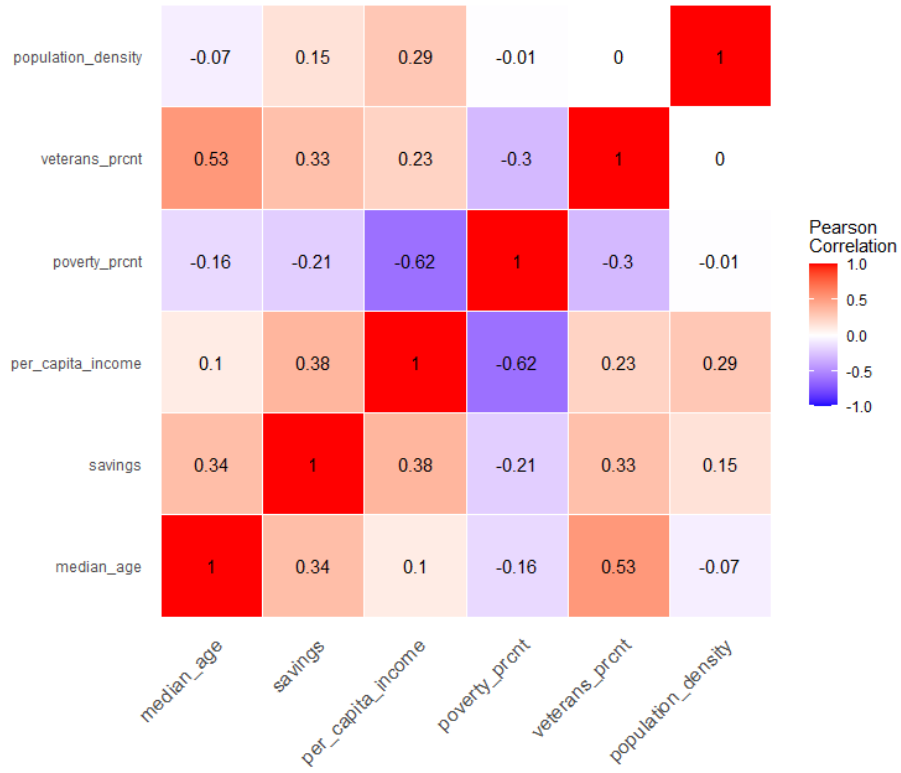


Figure 5: Correlation Heat Map for independent variables

Here, as all the independent variables are continuous, Pearson Correlation method has been used. For the output, it can be seen that there is not significant correlation between any 2 variables. The correlation between *poverty_prcnt* and *per_capita_income* is high. But, as this value at -0.62 is not high enough to reject any of these 2 variables, we will consider all these independent variables in this analysis.

Now that we have a better understanding of the data to be used, we will proceed for defining the Bayesian Regression Model.

3. Model Specification

As mentioned in the overview, we will implement the MCMC simulation for this Bayesian Regression Model using JAGS. The model can be given as below:

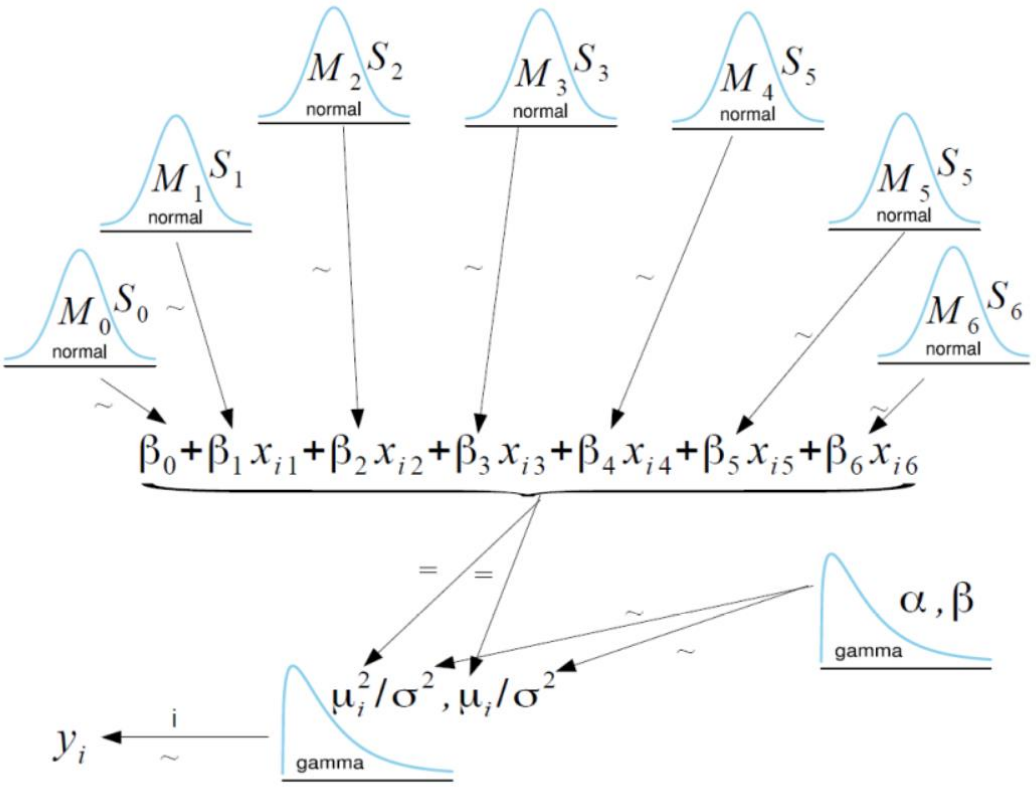


Figure 6: JAGS Model diagram

In the above model, y_i represents the posterior distribution of predicted *crime_index*.

Gamma(μ_i^2/σ^2, μ_i/σ^2) represents likelihood. Here, likelihood represents the data. We saw in the scatter plots that values for all the independent variables were positive. Also, we saw that normal-gamma model would be suitable here. So, we chose gamma model to represent the likelihood. To consider the mean(μ) and variance(σ) in gamma model, we performed re-parameterization. We used the first parameter of gamma distribution as μ_i^2/σ^2 and the second parameter of gamma distribution as μ_i/σ^2. Here, mean(μ_i) will represent the predicted value and variance(σ^2) will represent the confidence level in the predicted value.

From above we know that variance(σ^2) will represent the level of confidence in estimated value. So, as a part of the Bayesian estimation, we will define a gamma prior distribution gamma(α, β) for variance(σ^2).

Now, as mean(μ_i) will represent the predicted value in gamma(μ_i^2/σ^2, μ_i/σ^2) distribution, using the above specified regression equation, mean(μ_i)can be given as:

$$\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_4 x_{4i} + \beta_5 x_{5i} + \beta_6 x_{6i} ,$$

where x_{1i}, x_{2i}, x_{3i}, x_{4i}, x_{5i} and x_{6i} represent the data for variables *median_age*, *savings*, *per_capita_income*, *poverty_prcnt*, *veterans_prcnt* and *population_density* respectively. Here, as along with the prediction y_i we are trying to find the Bayesian estimate for β_0, β_1, β_2, β_3, β_4, β_5 and β_6, we will define a distribution for each of them. These distributions will also represent the prior information of how various parameters affect the crime index.

As β_0 is the intercept and β_1, β_2, β_3, β_4, β_5, and β_6 are the coefficients of a regression equation, they can take any value from $(-\infty, +\infty)$. So, we will use normal distribution to represent these priors. Moreover, the mean and variance for each of these prior distributions will help us specify the expert knowledge.

So, the prior distributions for β_0, β_1, β_2, β_3, β_4, β_5, and β_6 can be given as:

$\beta_0 \sim normal(M_0, S_0)$

$\beta_1 \sim normal(M_1, S_1)$

$\beta_2 \sim normal(M_2, S_2)$

$\beta_3 \sim normal(M_3, S_3)$

$\beta_4 \sim normal(M_4, S_4)$

$\beta_5 \sim normal(M_5, S_5)$

$\beta_6 \sim normal(M_6, S_6)$

Here, M_0 to M_6 are the corresponding mean values and S_0 to S_6 are the corresponding variance values. We will use these values to specify our belief as prior information for each parameter.

So, above mentioned is our model for the Bayesian regression. While specifying this model in JAGS, we need to specify 2 parts. First part will specify how to standardize the data. Second part will specify our actual model. The first and second part will altogether form the model string for our model. The model string can be found in Appendix section.

After defining the model, we will specify the prior information for coefficients of all the parameters. We will also check the sensitivity of these prior distributions on the corresponding posterior distributions.

4. Prior Specification

Here, we will specify the prior distributions and observe its effect on the corresponding posterior distributions.

The prior information for this analysis has been derived by performing a simple multiple regression on this data using MS Excel. The intercept and co-efficient values have been taken from these results and the belief in these values (variance) has been decided by our own understanding of how different factors of society affect the crime.

We will start by specifying the prior distribution for variance(σ^2). We will then specify the prior distribution for the intercept β_0. Next, we will specify the prior distribution for the coefficient of each

parameter. β_1, β_2, β_3, β_4, β_5, and β_6 are the coefficients for parameters *median_age*, *savings*, *per_capita_income*, *poverty_prcnt*, *veterans_prcnt* and *population_density* respectively.

While specifying the prior distributions and computing their respective posterior distributions, we will use a random sample of 270 data points from the available data of 2,702 data points. Also, the values of MCMC parameters are set as below:

Adaptation Steps: 1000

Burn-in Steps: 500

Chains: 3

Thinning Steps: 135

Saved Steps: 2100

The specification of prior distributions are as below:

4.1. For variance σ^2

As seen in the model specification, the prior distribution for σ^2 is:

$\sigma^2 \sim$ gamma(α, β)

where α represents first parameter of the distribution and β represents second parameter of the distribution. Here, we will specify this distribution as non-informative by setting the value of both the parameters to 0.01

So, for this prior specification, the posterior distribution for σ^2 is as below:

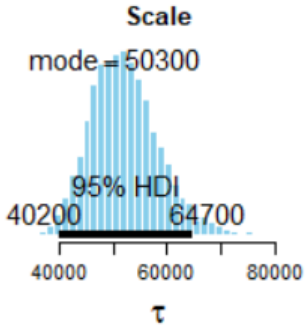


Figure 7: Posterior distribution of variance(σ^2)

From above output, we can see that the Bayesian estimate of σ^2 is 50300 with a 95% High Density Interval(HDI) of [40200, 64700].

(P.T.O.)

4.2. For intercept β_0

As seen in the model specification, the prior distribution for β_0 is:

$$\beta_0 \sim \text{normal}(M_0, S_0)$$

where M_0 represents mean of the distribution and S_0 represents variance of the distribution. As β_0 is the intercept of regression model, it is not associated with any of the parameters. So, it does not represent any prior information. For this reason, we will specify this distribution as non-informative. Here, the mean(M_0) will be zero and the variance(S_0) would be 40. The variance here will not be standardized and will represent a non-informative prior.

So, for this prior specification, the posterior distribution for β_0 is as below:

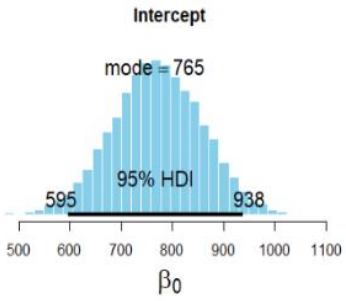


Figure 8: Posterior distribution for β_0

From above output, we can see that the Bayesian estimate of β_0 is 765 with a 95% High Density Interval(HDI) of [595, 938].

4.3. For coefficient β_1 of parameter *median_age*

As seen in the model specification, the prior distribution for β_1 is,

$$\beta_1 \sim \text{normal}(M_1, S_1)$$

where M_1 represents mean of the distribution and S_1 represents variance of the distribution. The prior information which we have for *median_age* parameter is that for every unit increase in the median age of county, the crime index for that county will decrease by 9 units. We have a high belief in this knowledge.

The mean(M_1) for this prior distribution would be -9. As the level of belief is high, the variance(S_1) would be 4.5. Low value of variance represents a strong belief in the information.

So, for this prior specification, the posterior distribution of β_1 is as below:

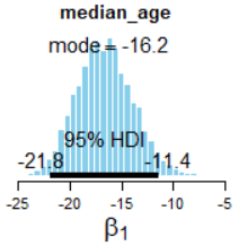


Figure 9: Posterior distribution for β_1

From above output, we can see that the Bayesian estimate of β_1 is -16.2 with a 95% High Density Interval(HDI) of [-21.8, -11.4]. To check the sensitivity of above posterior distribution, we will change the variance(S_1) from 4.5 to 2.7. This means, we will make our belief in the prior information stronger.

So, for the updated prior specification, the posterior distribution of β_1 is as below:

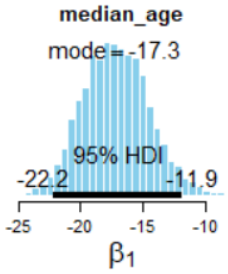


Figure 10: Posterior distribution for β_1 with updated prior information

From the above output, for change in variance(S_1) from 4.5 to 2.7, the Bayesian estimate for β_1 changed from -16.2 to -17.3. Also, the 95% HDI change from [-21.8, -11.4] to [-22.2, -11.9].

4.4. For coefficient β_2 of parameter *savings*

As seen in the model specification, the prior distribution for β_2 is,

$$\beta_2 \sim \text{normal}(M_2, S_2)$$

where M_2 represents mean of the distribution and S_2 represents variance of the distribution. The prior information which we have for *savings* parameter is that the average savings of people in the county will have no effect on the crime index. We have a high belief in this knowledge.

As the prior suggests that there would be no effect of *savings* on the *crime_index*, the mean(M_2) for this prior distribution would be 0. As the level of belief is high, the variance(S_2) would be 4.5. Low value of variance represents a strong belief in the information.

So, for this prior specification, the posterior distribution of β_2 is as below:

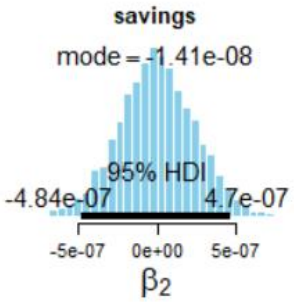


Figure 11: Posterior distribution for β_2

From above output, we can see that the Bayesian estimate of β_2 is -0.00000141 with a 95% High Density Interval(HDI) of [-0.0000484, 0.0000047]. To check the sensitivity of above posterior distribution, we will change the variance(S_2) from 4.5 to 2.7. This means, we will make our belief in the prior information stronger.

So, for the updated prior specification, the posterior distribution of β_2 is as below:

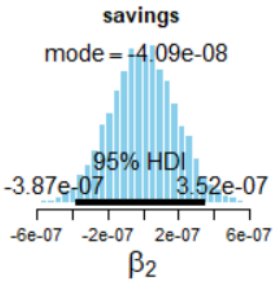


Figure 12: Posterior distribution for β_2 with updated prior information

From the above output, for change in variance(S_2) from 4.5 to 2.7, the Bayesian estimate for β_2 changed from -0.00000141 to -0.00000409. Also, the 95% HDI change from [-0.0000484, 0.0000047] to [-0.0000387, 0.0000352].

4.5. For coefficient β_3 of parameter *per_capita_income*

As seen in the model specification, the prior distribution for β_3 is,

$$\beta_3 \sim \text{normal}(M_3, S_3)$$

where M_3 represents mean of the distribution and S_3 represents variance of the distribution. The prior information which we have for *per_capita_income* parameter is that for every 10,00,000 units of increase in per capita income of the county, the crime index will increase by 1 unit. We have a high belief in this knowledge.

As the scale of data is in 1 units, the mean(M_3) for this prior distribution would be 0.000001. As the level of belief is high, the variance(S_3) would be 4.5. Low value of variance represents a strong belief in the information.

So, for this prior specification, the posterior distribution of β_3 is as below:

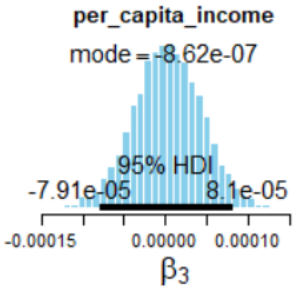


Figure 13: Posterior distribution for β_3

From above output, we can see that the Bayesian estimate of β_3 is -0.0000862 with a 95% High Density Interval(HDI) of [-0.00791, 0.00081]. To check the sensitivity of above posterior distribution, we will change the variance(S_3) from 4.5 to 2.7. This means, we will make our belief in the prior information stronger.

So, for the updated prior specification, the posterior distribution of β_3 is as below:

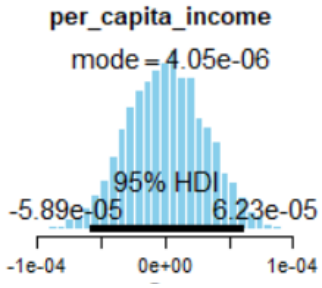


Figure 14: Posterior distribution for β_3 with updated prior information

From the above output, for change in variance(S_3) from 4.5 to 2.7, the Bayesian estimate for β_3 changed from -0.0000862 to 0.000405. Also, the 95% HDI change from [-0.00791, 0.00081] to [-0.00589, 0.00623].

4.6. For coefficient β_4 of parameter *poverty_prcnt*

As seen in the model specification, the prior distribution for β_4 is,

$$\beta_4 \sim \text{normal}(M_4, S_4)$$

where M_4 represents mean of the distribution and S_4 represents variance of the distribution. The prior information which we have for *poverty_prcnt* parameter is that for every 10 percent increase in population below poverty line, the crime index will increase by 8 units. We have a very high belief in this knowledge.

As the scale of data is in 1 units, the mean(M_4) for this prior distribution would be 0.8. As the level of belief is very high, the variance(S_4) would be 1.2. Very low value of variance represents a very strong belief in the information.

So, for this prior specification, the posterior distribution of β_4 is as below:

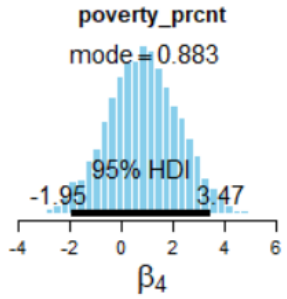


Figure 15: Posterior distribution for β_4

From above output, we can see that the Bayesian estimate of β_4 is 0.883 with a 95% High Density Interval(HDI) of [-1.95, 3.47]. To check the sensitivity of above posterior distribution, we will change the variance(S_4) from 1.2 to 0.3. This means, we will make our belief in the prior information stronger.

So, for the updated prior specification, the posterior distribution of β_4 is as below:

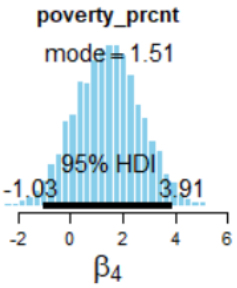


Figure 16: Posterior distribution for β_4 with updated prior information

From the above output, for change in variance(S_4) from 1.2 to 0.3, the Bayesian estimate for β_4 changed from 0.883 to 1.51. Also, the 95% HDI change from [-1.95, 3.47] to [-1.03, 3.91].

4.7. For coefficient β_5 of parameter *veterans_prcnt*

As seen in the model specification, the prior distribution for β_5 is,

$$\beta_5 \sim \text{normal}(M_5, S_5)$$

where M_5 represents mean of the distribution and S_5 represents variance of the distribution. The prior information which we have for *veterans_prcnt* parameter is that for every 1 percent increase of veterans in the population of county, the crime index decreases by 4 units. We have a weak belief in this knowledge.

The mean(M_5) for this prior distribution would be -4. As the level of belief is low, the variance(S_5) would be 12. High value of variance represents a weak belief in the information.

So, for this prior specification, the posterior distribution of β_5 is as below:

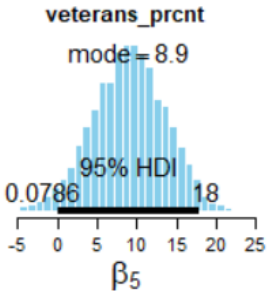


Figure 17: Posterior distribution for β_5

From above output, we can see that the Bayesian estimate of β_5 is 8.9 with a 95% High Density Interval(HDI) of [0.0786, 18]. To check the sensitivity of above posterior distribution, we will change the variance(S_5) from 12 to 15. This means, we will make our belief in the prior information weaker.

(P.T.O.)

So, for the updated prior specification, the posterior distribution of β_5 is as below:

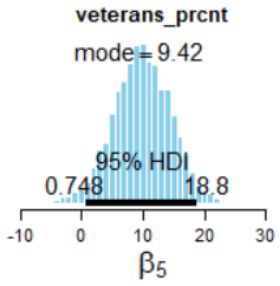


Figure 18: Posterior distribution for β_5 with updated prior information

From the above output, for change in variance(S_5) from 12 to 15, the Bayesian estimate for β_5 changed from 8.9 to 9.42. Also, the 95% HDI change from [0.0786, 18] to [0.748, 18.8].

4.8. For coefficient β_6 of parameter *population_density*

As seen in the model specification, the prior distribution for β_6 is,

$$\beta_6 \sim \text{normal}(M_6, S_6)$$

where M_6 represents mean of the distribution and S_6 represents variance of the distribution. The prior information which we have for *population_density* parameter is that for every 10,000 units increase in population density of the county, the crime index will increase by 1 unit. We have a high belief in this knowledge.

As the scale of data is in 1 units, the mean(M_6) for this prior distribution would be 0.0001. As the level of belief is high, the variance(S_6) would be 4.5. Low value of variance represents a strong belief in the information.

So, for this prior specification, the posterior distribution of β_6 is as below:

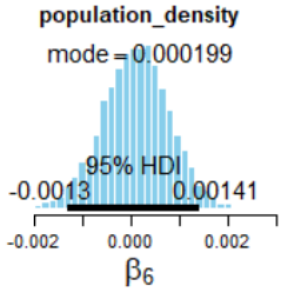


Figure 19: Posterior distribution for β_6

From above output, we can see that the Bayesian estimate of β_6 is 0.000199 with a 95% High Density Interval(HDI) of [-0.0013, 0.00141]. To check the sensitivity of above posterior distribution, we will change the variance(S_6) from 4.5 to 2.7. This means, we will make our belief in the prior information stronger.

(P.T.O.)

So, for the updated prior specification, the posterior distribution of β_6 is as below:

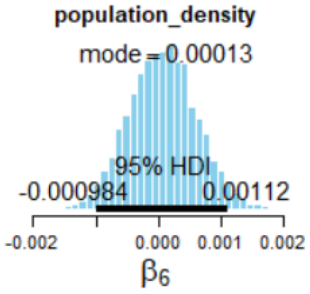


Figure 20: Posterior distribution for β_6 with updated prior information

From the above output, for change in variance(S_6) from 4.5 to 2.7, the Bayesian estimate for β_6 changed from 0.000199 to 0.00013. Also, the 95% HDI change from [-0.0013, 0.00141] to [-0.000984, 0.00112].

5. MCMC Diagnostics

After running the MCMC simulation to find the Bayesian estimates, it is important to check the generated chains for their representativeness and accuracy. We also need check for the efficiency. Here, we will check the MCMC diagnostics of the chains generated for the intercept(β_0), coefficients(β_1, β_2, β_3, β_4, β_5 and β_6) and the variance(σ^2). We will assess them for their representativeness and accuracy. To improve the representativeness and accuracy wherever required, we will tune the MCMC parameters like burn-in steps, number of chains, number of thinning steps and number of saved steps.

As we had considered a sample of 270 to obtain the posterior distributions, we will again consider the same sample of 270 for assessing MCMC diagnostics.

5.1. Initial settings

We will first check the diagnostics for MCMC chains generated by below MCMC parameters:

Adaptation Steps: 1000

Burn-in Steps: 500

Chains: 3

Thinning Steps: 81

Saved Steps: 2100

(P.T.O.)

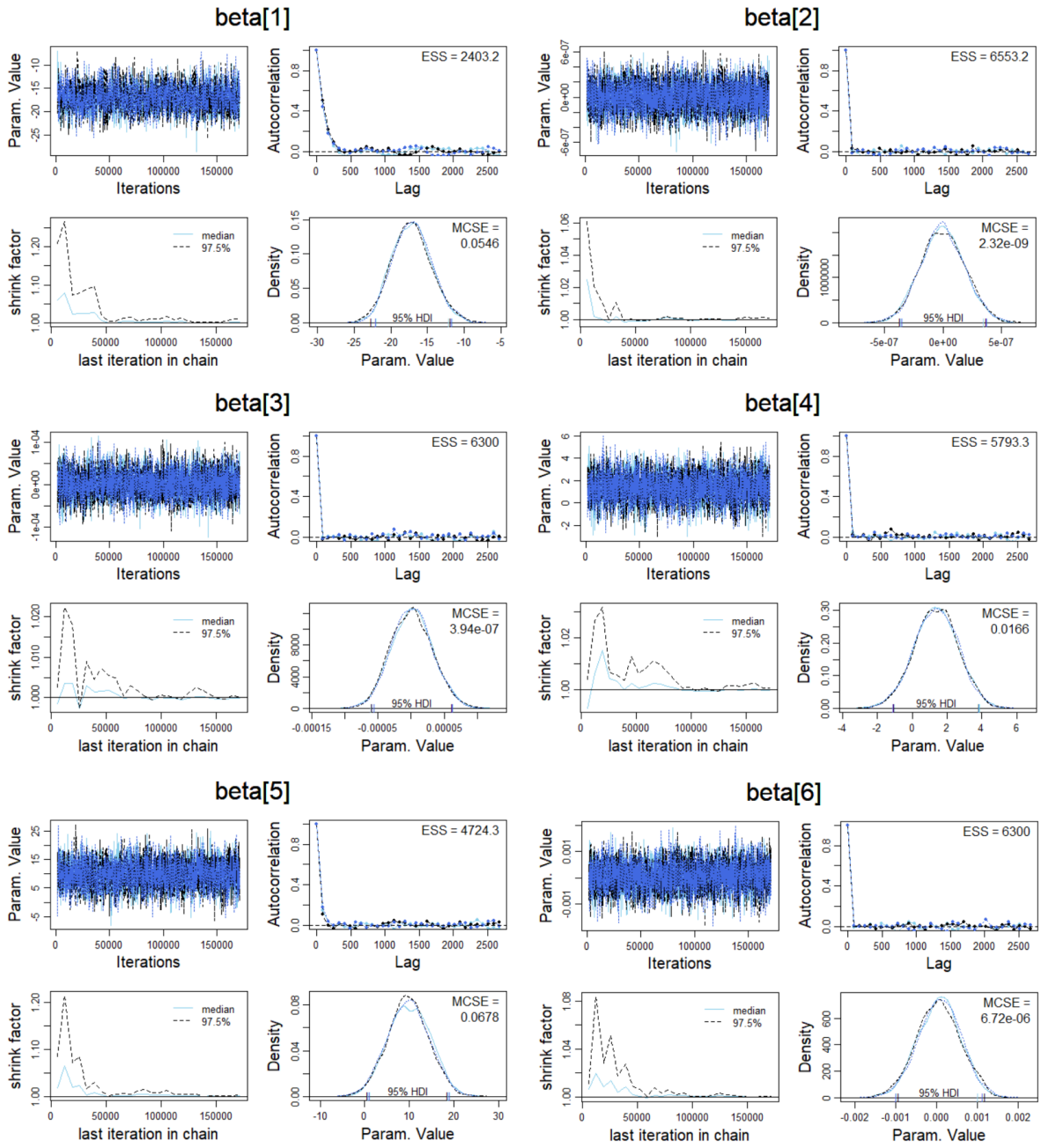


Figure 21: MCMC Diagnostic Checks for β_1, β_2, β_3, β_4, β_5, and β_6

First, we will check the representativeness of the MCMC chains for the coefficients. The shrink factor should be less than 1.2. From the above output, the shrink factor for chains of all coefficients except β_1 and β_5, is under 1.2. Next, all the chains converge at the mean level and mix well. Also, the density distributions for almost all the parameters overlap well. Coming to the autocorrelation, it should be near to zero. However, except for β_1 and β_5, there is autocorrelation present in the MCMC chains. This can be fixed by increasing the number of thinning steps.

Next, we will check the accuracy of the MCMC chains. From the above output, Estimated Sample Size (ESS) for all the coefficients are acceptable as these values range in between 2000 to 6300 for all the

coefficients. Also, the Monte Carlo Standard Error(MCSE) for all the parameters is pretty good as all the values are below 0.

So, overall, the MCMC chains for above parameters setting are acceptable in terms of accuracy but not so good in terms of representativeness for β_1 and β_5. To eliminate the issues mentioned above, we will change the values of MCMC parameter Number of Thinning Steps.

5.2. Updated settings

We will increase the Number of Thinning Steps from 81 to 135. This will help us decrease the autocorrelation for β_1 and β_5. So, the diagnostic check for MCMC chains generated for coefficients β_1, β_2, β_3, β_4, β_5 and β_6 using updated MCMC parameters is as below:

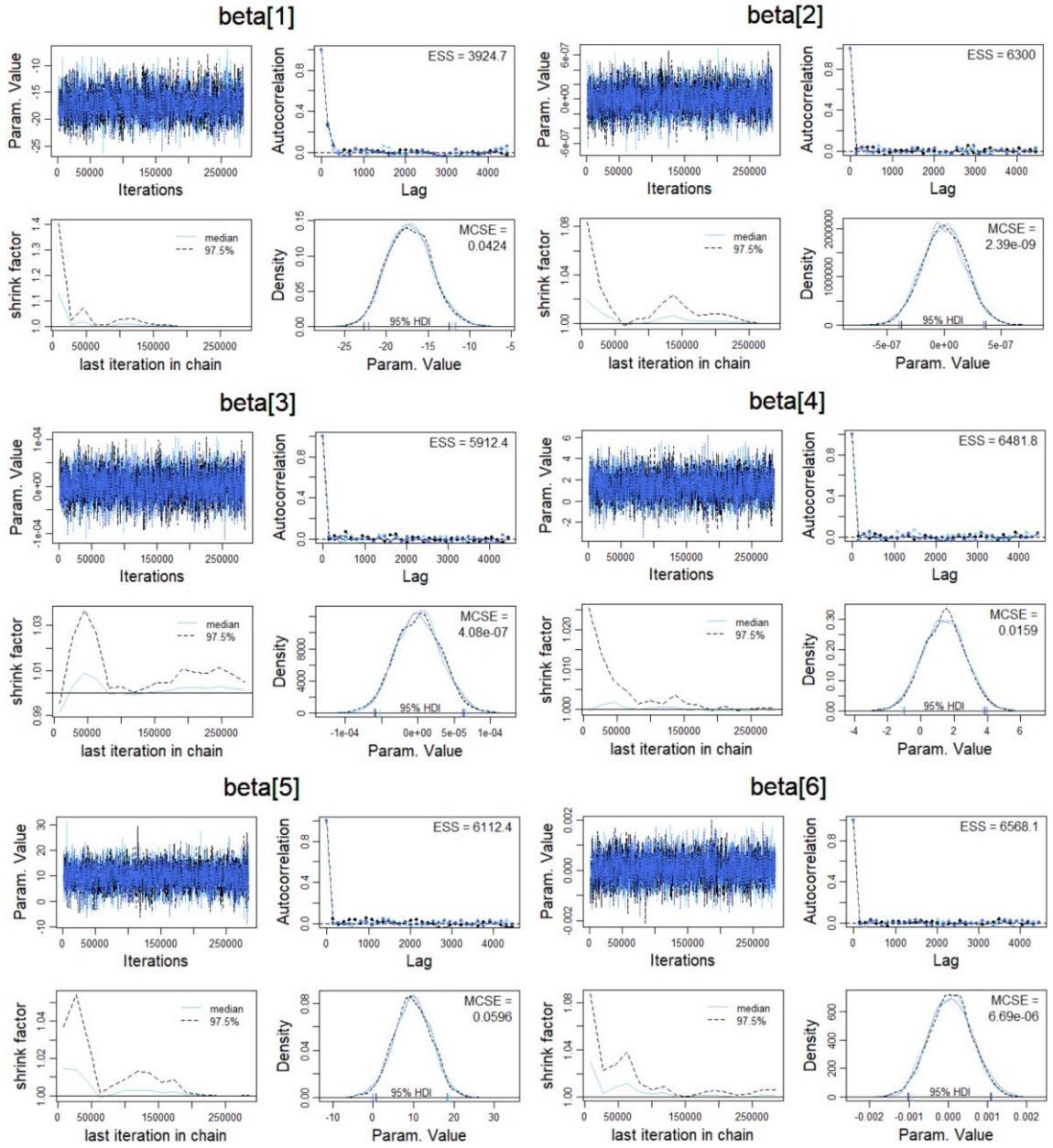


Figure 22: MCMC Diagnostic Checks for β_1, β_2, β_3, β_4, β_5, and β_6 with updated MCMC settings

From the above output, in terms of representativeness, the autocorrelation for all the parameters now seems very close to 0. It can still be brought closer to zero for β_1, but it is acceptable. The shrink factor too is below 1.2 now. From the above output, we can also see that by updating the MCMC parameters settings, other factors in the diagnostic check have not been affected negatively. For all the parameters, the chains converge at the mean level and mix well, the shrink factor is below 1.2, the distributions for all the parameters overlap well, the ESS for all the parameters is in acceptable range, and the MCSE is well below 0 for all the parameters.

So, for below MCMC parameter settings,

Adaptation Steps: 1000

Burn-in Steps: 500

Chains: 3

Thinning Steps: 135

Saved Steps: 2100

we got good results out of MCMC diagnostic checks for chains of coefficients β_1, β_2, β_3, β_4, β_5, and β_6.

For above MCMC parameter settings, the MCMC diagnostics for chains generated while estimating variance(σ^2) and the intercept(β_0) are as below:

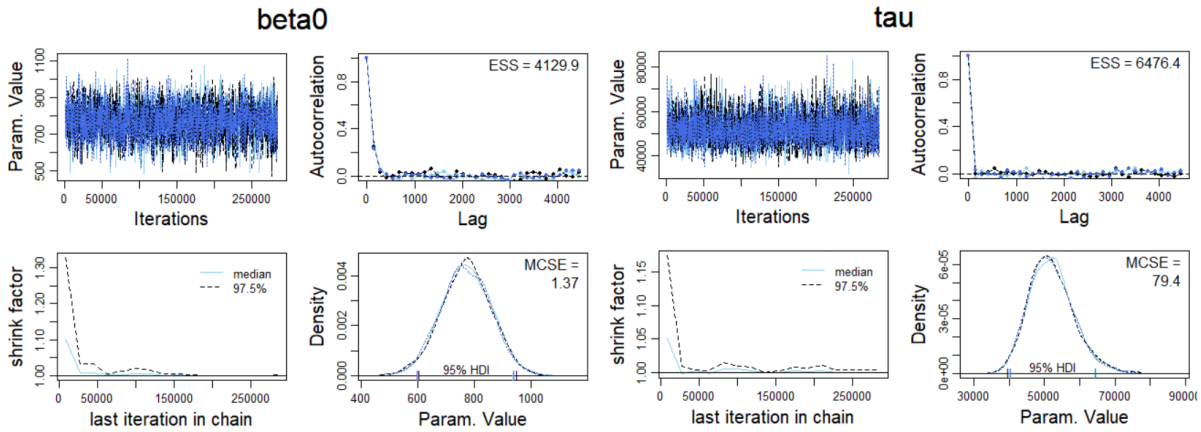


Figure 23: MCMC Diagnostic Checks for intercept(β_0) and variance(σ^2)

From the above output, in terms of representativeness, the shrink factor is good as it is less than 1.2 for σ^2. For β_0, it can still be reduced by increasing the number of MCMC chains. But as the shrink factor value for median is also below 1.2, it is acceptable. For both σ^2 and β_0, the chains converge at mean level and are mixed well. The distribution and respective HDI limits for all the chains overlap well. Also, the autocorrelation is very near to zero for σ^2. For β_0, the autocorrelation can be brought closer to zero, but this value too would be acceptable. In terms of accuracy, the ESS is acceptable at 4130 for σ^2 and 6476 for β_0.

The MCSE is high for σ^2 which can be reduced by tuning the MCMC parameters further. But due computational limitations, we will proceed with Bayesian estimate for variance(σ^2) and note that this value can be problematic.

Now, in terms of efficiency, the parallel run completed in lesser time compared to the sequential run. For various runs on a sample of 270 records, following run times were recorded for sequential and parallel runs:

Run	Adaptation Steps	Burn-in Steps	Chains	Thinning Steps	Saved Steps	Sequential Run	Parallel Run
1	1000	500	3	27	2100	**00:18:16**	**00:09:13**
2	1000	500	3	81	2100	**00:51:47**	**00:19:09**
3	1000	500	3	135	2100	**01:24:24**	**00:33:17**

Figure 24: Run-time comparison between sequential and parallel for various MCMC parameter settings

For the first run, MCMC simulation for the multiple linear regression model specified using JAGS ran nearly 50% faster. For run 2 and 3, the simulation ran around 40% faster.

The efficiency can be further improved by using the functionality of R packages like dclone, greta, and gputools. This has been discussed further in the Recommendations section.

6. Results

Using the best values for MCMC parameters and the best values for variance of prior distributions to represent the degree of belief, obtained previously, we run the MCMC estimation for the multiple linear regression model defined above. Here, we use the entire dataset with total of 2,702 data points. We will first obtain the Bayesian estimates for all the parameters(intercept, coefficients and variance) and then obtain the Bayesian estimates of predictions for *crime_index* using different values of predictors.

6.1. Bayesian Estimates for Parameters

Here, we will obtain the Bayesian estimates for variance(σ^2), intercept β_0, and the coefficients β_1, β_2, β_3, β_4, β_5, and β_6 for predictors *median_age*, *savings*, *per_capita_income*, *poverty_prcnt*, *veterans_prcnt* and *population_density* respectively. The prior distributions for all the parameters can be given as follows:

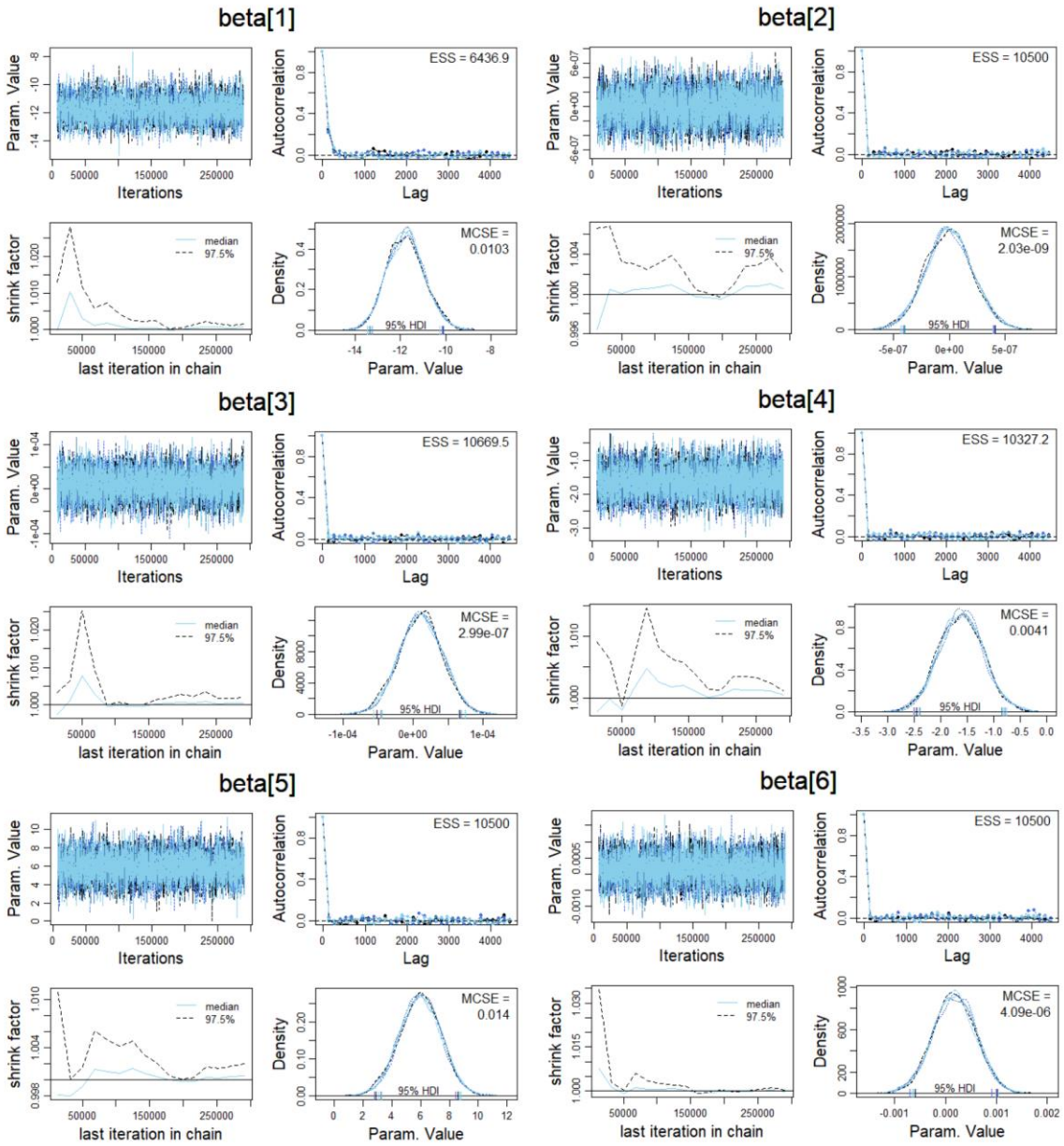
- σ^2 ~ gamma(0.01, 0.01)
- β_0 ~ normal(0, 40)
- β_1 ~ normal(-9, 2.7), standardized
- β_2 ~ normal(0, 2.7), standardized
- β_3 ~ normal(0.000001, 2.7), standardized
- β_4 ~ normal(0.8, 0.3), standardized
- β_5 ~ normal(-4, 15), standardized
- β_6 ~ normal(0.0001, 2.7), standardized

As we are considering all the data, we need to change the MCMC settings. The MCMC parameters settings can be given as:

- **Adaptation Steps:** 1000
- **Burn-in Steps:** 6000

- **Chains:** 5
- **Thinning Steps:** 135
- **Saved Steps:** 2100

After running the model using above mentioned prior distributions and the MCMC parameters settings, the MCMC diagnostics for all the above parameters are as below:



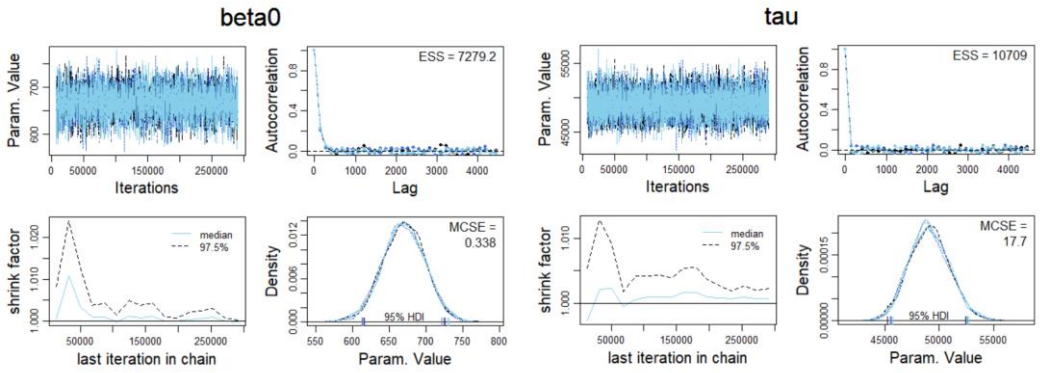


Figure 25: MCMC Diagnostic Checks for coefficients β_1, β_2, β_3, β_4, β_5, and β_6, intercept β_0 and variance σ^2

In terms of representativeness, the MCMC chains for all the parameters have shrink factor less than 1.2. All the chains converge at respective mean level and mix well. The density curves along with respective 95% HDIs of chains for all the parameters seem to nearly overlap. Also, the autocorrelation for chains of all the parameters is very close to 0. As discussed previously, for β_0 and β_1, the autocorrelation can be lowered further. So, in terms of representativeness, the generated MCMC chains are overall good.

In terms of accuracy, the MCMC chains for all the parameters have a good ESS of around 6000 to 10500. Also, the MCSE for chains of all the parameters except variance(σ^2) is well below 0. As for variance(σ^2), we have already noted in the MCMC diagnostics section that the Bayesian estimate for σ^2 will not be reliable. So, for other than variance(σ^2), in terms of accuracy, the generated MCMC chains are overall good.

After validating the MCMC chains for all the parameters, the respective posterior distributions for those parameters are as below:

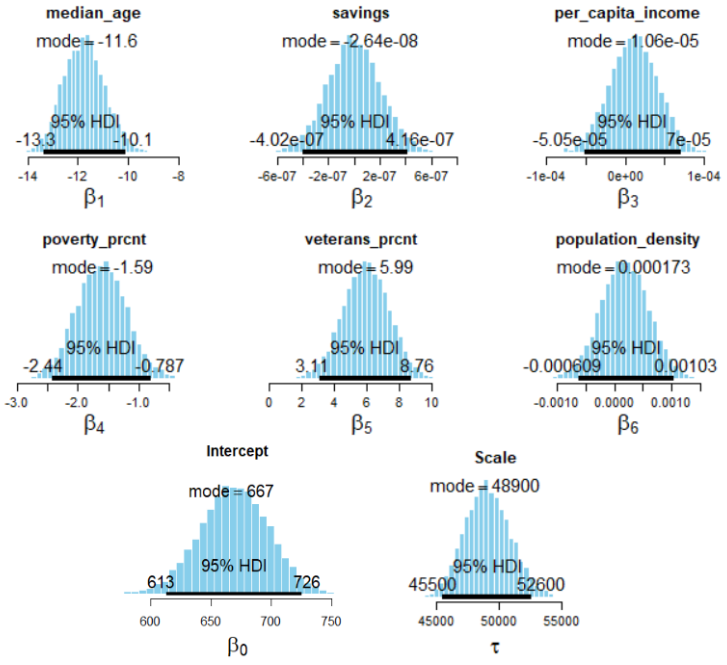


Figure 26: Posterior distributions for β_0, β_1, β_2, β_3, β_4, β_5, β_6, and variance σ^2

As seen in the above output, the Bayesian estimate along with the respective HDIs for all the parameters can be summarized as below:

- β_0 – Bayesian estimate of 667 with HDI of [613, 726]
- β_1 – Bayesian estimate of -11.6 with HDI of [-13.3, -10.1]
- β_2 – Bayesian estimate of -0.00000264 with HDI of [-0.0000402, 0.0000416]
- β_3 – Bayesian estimate of 0.00106 with HDI of [-0.00505, 0.00007]
- β_4 – Bayesian estimate of -1.59 with HDI of [-2.44, -0.787]
- β_5 – Bayesian estimate of 5.99 with HDI of [3.11, 8.76]
- β_6 – Bayesian estimate of 0.000173 with HDI of [-0.000609, 0.00103]
- σ^2 (tau) – Bayesian estimate of 48900 with HDI of [45500, 52600]

The interpretation of above values can be given as below:

- **β_0** – If none of the predictor value is given, the crime index for county would be 667.
- **β_1** – For every unit increase in the median age of people in the county, crime index will decrease by 11.6.
- **β_2** – For every dollar increase in the average savings of people in county, the crime index will decrease by 0.00000264. In other words, for every 100,000 dollars increase in the average savings of people in county, the crime index will increase by about 3.
- **β_3** – For every dollar increase in the per capita income of the county, the crime index will increase by 0.00106. In other words, for every 1000 dollar of increase in the per capita income, the crime index will increase by 1.
- **β_4** – For every percent increase in the proportion of people under poverty line for the county, crime index will decrease by around 2.
- **β_5** – For every percent increase in the proportion of veterans in the county, crime index will increase nearly by 6.
- **β_6** – For every unit increase of population density in the county, crime index will increase by 0.000173. In other words, for every 10,000 units increase in the population density, the crime index will increase nearly by 2.
- **σ^2 (tau)** – The distribution representing the predicted crime index will have a variance of 48,900.

In the above results, the Bayesian estimate of variance obtained is unusually high. As seen earlier in the MCMC diagnostics, this value is not reliable can depict incorrect value. Also, after obtaining the above estimates, it would be important to check which of the values are statistically significant and affect the behavior of *crime_index*.

To check which values are statistically significant, below hypothesis can be proposed.

H_0: Obtained value is not significant

H_A: Obtained value is significant.

For this hypothesis, the rejection rule can be defined as follows:

Reject: if 0 lies outside the 95% High Density Interval(HDI)

Fail to Reject: if 0 lies in between the 95% High Density Interval(HDI)

Now that we have defined the hypothesis and the decision rule, we will check if the values for intercept β_0 and coefficients β_1, β_2, β_3, β_4, β_5, and β_6 are significant.

6.1.1. Hypothesis for Intercept β_0

Considering the 95% HDI of [613, 726], 0 is well outside this range. So, we reject the null hypothesis and conclude that the Bayesian estimate for β_0 is **statistically significant**.

6.1.2. Hypothesis for coefficient β_1

Considering the 95% HDI of [-13.3, -10.1], 0 is well outside this range. So, we reject the null hypothesis and conclude that the Bayesian estimate for β_1 is **statistically significant**.

6.1.3. Hypothesis for coefficient β_2

Considering the 95% HDI of [-0.0000402, 0.0000416], 0 is on the upper edge of the interval. So, for 95% significance level, β_2 is **statistically insignificant**. However, as the HDI is for 95% significance level and 0 is on the edge of this interval, β_2 could be significant for any significance level below 94%.

6.1.4. Hypothesis for coefficient β_3

Considering the 95% HDI of [-0.00505, 0.00007], 0 is on the upper edge of the interval. So, for 95% significance level, β_3 is **statistically insignificant**. However, as the HDI is for 95% significance level and 0 is on the edge of this interval, β_3 could be significant for any significance level below 94%.

6.1.5. Hypothesis for coefficient β_4

Considering the 95%HDI of [-2.44, -0.787], 0 is outside this range. So, we reject the null hypothesis and conclude that the Bayesian estimate for β_4 is **statistically significant**.

6.1.6. Hypothesis for coefficient β_5

Considering the 95%HDI of [3.11, 8.76], 0 is outside this range. So, we reject the null hypothesis and conclude that the Bayesian estimate for β_5 is **statistically significant**.

6.1.4. Hypothesis for coefficient β_6

Considering the 95% HDI of [-0.000609, 0.00103], 0 is on the upper edge of the interval. So, for 95% significance level, β_6 is **statistically insignificant**. However, as the HDI is for 95% significance level and 0 is on the edge of this interval, β_6 could be significant for any significance level below 94%.

6.2. Bayesian Estimates for Predictions

Now that we have estimated the parameters, we will predict the crime index for below values of *median_age*, *savings*, *per_capita_income*, *poverty_prcnt, veterans_prcnt* and *population_density* of hypothetical counties:

(P.T.O.)

median_age	savings	per_capita_income	poverty_prcnt	veterans_prcnt	population_density
33	87000	50000	7.02	10	200
18	30000	15000	17.19	19.22	1
60	690000	4300	50.89	29	35430
42	50000	26700	3.6	4.58	8080.59
50	400000	16300	34.77	7.77	3000
27	10800	12200	5.2	18.59	131

The R-squared value and predictions obtained for above values of the predictors are as below:

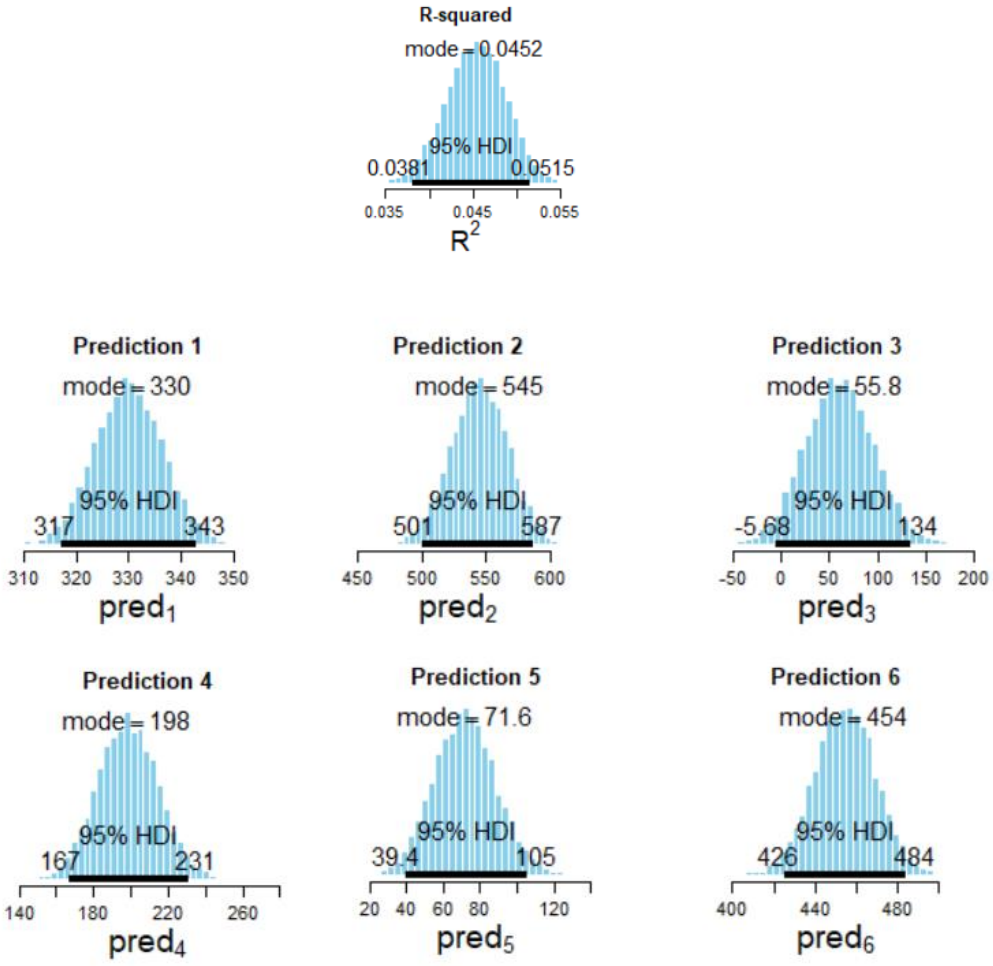


Figure 27: Posterior distributions for predictions and R-squared value

The above results can be summarized as below:

- For a county with median_age = 33, savings = 87,000, per_capita_income = 50,000, poverty_prcnt = 7.02, veterans_prcnt = 10,and population_density = 200, the crime index will be 330. Also, there is 95% chance that the crime index would lie in between 317 and 343.

- For a county with median_age = 18, savings = 30,000, per_capita_income = 15,000, poverty_prcnt = 17.19, veterans_prcnt = 19.22,and population_density = 1, the crime index will be 545. Also, there is 95% chance that the crime index would lie in between 501 and 587.
- For a county with median_age = 60, savings = 6,90,000, per_capita_income = 4,300, poverty_prcnt = 50.89, veterans_prcnt = 29,and population_density = 35,430, the crime index will be 55.8 (56). Also, there is 95% chance that the crime index would lie in between -5.68 and 134.
- For a county with median_age = 42, savings = 50,000, per_capita_income = 26,700, poverty_prcnt = 3.6, veterans_prcnt = 4.58,and population_density = 8,080.59, the crime index will be 198. Also, there is 95% chance that the crime index would lie in between 167 and 231.
- For a county with median_age = 50, savings = 4,00,000, per_capita_income = 16,300, poverty_prcnt = 34.77, veterans_prcnt = 7.77,and population_density = 3,000, the crime index will be 71.6 (72). Also, there is 95% chance that the crime index would lie in between 39.4 and 105.
- For a county with median_age = 27, savings = 10,800, per_capita_income = 12,200, poverty_prcnt = 5.2, veterans_prcnt = 18.59,and population_density = 131, the crime index will be 454. Also, there is 95% chance that the crime index would lie in between 426 and 484.

The R-squared value for this multiple linear regression model is 0.0452. This means that the model covers just 4.52% of the variation in the data. However, in the Bayesian version of linear regression, R-squared value is not a reliable measure. This is because the model takes into consideration even the expert knowledge in form of prior information.

Lastly, we will now plot a kernel density estimate for observed crime index values and the crime index values predicted by the model.

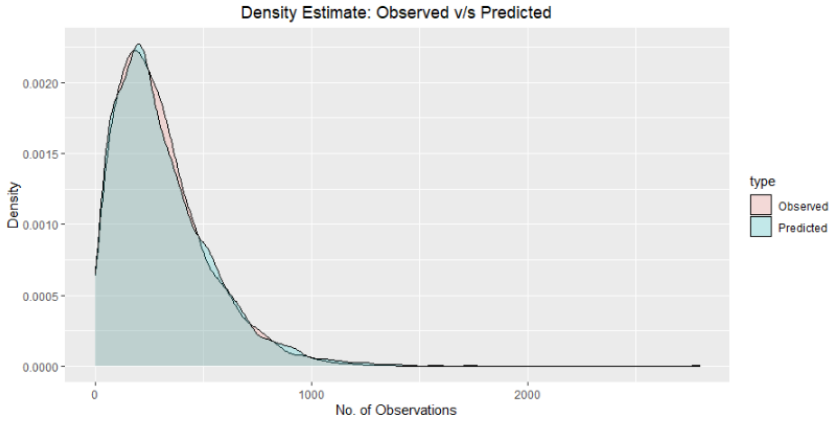


Figure 28: Kernel Density Estimate: Observed v/s Predicted

From the above plot and as suggested previously, the R-squared value is not an effective measure to assess the goodness of fit for linear regression model. Looking at the above plot, behavior of the *crime_index* is well explained by the model. However, this can be improved further by multiple ways. Few of such methods are discussed in the Recommendations section.

7. Conclusion

In this analysis, using Bayesian approach, the multiple linear regression model obtained was as below:

crime_index = 667 -

11.6*median_age - 0.00000264*savings + 0.00106*per_capita_income -

1.59*poverty_prcnt + 5.99*veterans_prcnt + 0.000173*population_density

So, using the above equation, crime index for a particular county be explained using the demographic and economic information for that county.

8. Recommendations

8.1. Efficiency

The efficiency of MCMC simulations can be improved in following ways:

- By using the parallel functionality provided by the dclone R package. The function that can be used is jags.parfit().
- By utilizing the GPUs. This can be done by writing custom functions to run the MCMC simulation and utilizing the GPUs using R packages like gputools and cudaBayesreg.
- Another package that helps utilization of GPUs is greta. Implementing the MCMC simulations using greta can help in utilizing the GPUs for processing.

8.2. Goodness of Fit

The goodness of fit for the model can be improved in following ways:

- Considering a bigger sample of data.
- Considering normalizing the predictors by using different transformation like log, Box-Cox, etc.
- Considering handling the outlying values.
- Tuning the MCMC parameters further until all the boxes for diagnostic check are ticked.

(P.T.O.)

9. References

- Prof. Haydar Demirhan, MATH2269 Module 5 notes – Just Another Gibbs Sampler - JAGS, School of Science, RMIT university.
- Prof. Haydar Demirhan, MATH2269 Module 6 notes – Bayesian Linear Regression, School of Science, RMIT university.
- Aditya Kakde, MATH2269 Assignment 2, Implementing Multiple Linear Regression model to predict Property Sales Prices in Melbourne using Bayesian Statistics.
- County level results for percent voting for Bill Clinton in 1992 Presidential Election and Demographic variables, U.S. Census Bureau, http://users.stat.ufl.edu/~winner/data/clinton1.dat, accessed on 3-Oct-20.
- R codes adopted from DBDA2Eprograms.zip by Kruschke, J. K. (2015). Doing Bayesian Data Analysis, Second Edition: A Tutorial with R, JAGS, and Stan. Academic Press / Elsevier.
- ggplot2 : Quick correlation matrix heatmap - R software and data visualization, STHDA, http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visualization, accessed on 5-Oct-20.
- Patric Zhao(August 4, 2014), Accelerate R Applications with CUDA, https://developer.nvidia.com/blog/accelerate-r-applications-cuda/#:~:text=The%20GPU%2DAccelerated%20R%20Software,including%20C%2C%20C%2B%2B%20and%20Fortran, accessed on 30-Sep-20.
- Peter Solymos (2010). dclone: Data Cloning in R. The R Journal 2(2), 29-37. https://journal.r-project.org/.
- Getting started with greta, https://greta-stats.org/articles/get_started.html, accessed on 2-Oct-20.

10. Appendix

1. Code for setting the stage

```
# set the environment
graphics.off()
rm(list = ls())

# import packages
library(readr)
library(dplyr)
library(reshape2)
library(ggplot2)
library(ggpubr)
library(corrplot)
library(ks)
library(rjags)
library(runjags)

# import MCMC utilities
source("DBDA2E-utilities.R")
```

(P.T.O.)

2. Code for function to find MCMC summary

```
# FUNCTION TO FIND MCMC SUMMARY

summary_MCMC = function(codaSamples, compVal = NULL) {
  summaryInfo = NULL
  mcmcMat = as.matrix(codaSamples, chains = TRUE)
  paramName = colnames(mcmcMat)
  for (pName in paramName) {
    if (pName %in% colnames(compVal)) {
      if (!is.na(compVal[pName])) {
        summaryInfo = rbind(summaryInfo,
                            summarizePost(
                              paramSampleVec = mcmcMat[, pName],
                              compVal = as.numeric(compVal[pName])
                            ))
      }
      else {
        summaryInfo = rbind(summaryInfo,
                            summarizePost(paramSampleVec = mcmcMat[,
pName]))
      }
    } else {
      summaryInfo = rbind(summaryInfo,
                          summarizePost(paramSampleVec = mcmcMat[, pName]))
    }
  }
  rownames(summaryInfo) = paramName

  return(summaryInfo)
}
```

3. Code for function to plot the MCMC results

```
plot_MCMC = function(codaSamples,
                     data,
                     xName = "x",
                     yName = "y",
                     showCurve = FALSE,
                     compVal = NULL,
                     saveName = NULL,
                     saveType = "jpg") {
  y = data[, yName]
  x = as.matrix(data[, xName])

  mcmcMat = as.matrix(codaSamples, chains = TRUE)
  chainLength = NROW(mcmcMat)
  zbeta0 = mcmcMat[, "zbeta0"]
  zbeta  = mcmcMat[, grep("^zbeta$|^zbeta\\[", colnames(mcmcMat))]

  if (ncol(x) == 1) {
    zbeta = matrix(zbeta, ncol = 1)
  }

  zVar = mcmcMat[, "zVar"]
  beta0 = mcmcMat[, "beta0"]
  beta  = mcmcMat[, grep("^beta$|^beta\\[", colnames(mcmcMat))]
```

```
  if (ncol(x) == 1) {
    beta = matrix(beta, ncol = 1)
  }

  tau = mcmcMat[, "tau"]

  pred  = mcmcMat[, grep("^pred$|^pred\\[", colnames(mcmcMat))]

  if (ncol(x) == 1) {
    pred = matrix(pred, ncol = 1)
  }

  #-----------------

  # Compute R-squared value

  Rsq = zbeta %*% matrix(cor(y, x), ncol = 1)

  #-----------------

  # Marginal histograms

  plot_hist = function(panelCount,
                       saveName,
                       finished = FALSE,
                       nRow = 2,
                       nCol = 3) {

    # If finishing a set:
    if (finished == TRUE) {
      if (!is.null(saveName)) {
        saveGraph(file = paste0(saveName, ceiling((
          panelCount - 1
        ) / (nRow * nCol))),
        type = saveType)
      }
      panelCount = 1 # re-set panelCount
      return(panelCount)
    } else {
      # If this is first panel of a graph:
      if ((panelCount %% (nRow * nCol)) == 1) {
        # If previous graph was open, save previous one:
        if (panelCount > 1 & !is.null(saveName)) {
          saveGraph(file = paste0(saveName, (panelCount %/% (nRow *
nCol))),
                    type = saveType)
        }

        # Open new graph
        openGraph(width = nCol * 7.0 / 3, height = nRow * 2.0)
        layout(matrix(1:(nRow * nCol), nrow = nRow, byrow = TRUE))
        par(mar = c(4, 4, 2.5, 0.5),
            mgp = c(2.5, 0.7, 0))
      }

      # Increment and return panel count:
      panelCount = panelCount + 1
      return(panelCount)
    }
```

```
  }

  #-----------------

  # Original scale:

  panelCount = 1
  if (!is.na(compVal["beta0"])) {
    panelCount = plot_hist(panelCount , saveName = paste0(saveName,
"PostMarg_beta0"))
    histInfo = plotPost(
      beta0,
      cex.lab = 1.75,
      showCurve = showCurve,
      xlab = bquote(beta[0]),
      main = "Intercept",
      compVal = as.numeric(compVal["beta0"])
    )
  } else {
    histInfo = plotPost(
      beta0,
      cex.lab = 1.75,
      showCurve = showCurve,
      xlab = bquote(beta[0]),
      main = "Intercept"
    )
  }

  for (bIdx in 1:ncol(beta)) {
    panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMarg_", bIdx))
    if (!is.na(compVal[paste0("beta[", bIdx, "]")])) {
      histInfo = plotPost(
        beta[, bIdx],
        cex.lab = 1.75,
        showCurve = showCurve,
        xlab = bquote(beta[.(bIdx)]),
        main = xName[bIdx],
        compVal = as.numeric(compVal[paste0("beta[", bIdx, "]")])
      )
    } else {
      histInfo = plotPost(
        beta[, bIdx],
        cex.lab = 1.75,
        showCurve = showCurve,
        xlab = bquote(beta[.(bIdx)]),
        main = xName[bIdx]
      )
    }
  }

  panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMarg_scale"))
  histInfo = plotPost(
    tau,
    cex.lab = 1.75,
    showCurve = showCurve,
    xlab = bquote(tau),
    main = paste("Scale")
```

```
  )

  panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMarg_rsq"))
  histInfo = plotPost(
    Rsq,
    cex.lab = 1.75,
    showCurve = showCurve,
    xlab = bquote(R ^ 2),
    main = paste("R-squared")
  )

  for (predIdx in 1:ncol(pred)) {
    panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMarg_", predIdx))
    histInfo = plotPost(
      pred[, predIdx],
      cex.lab = 1.75,
      showCurve = showCurve,
      xlab = bquote(pred[.(predIdx)]),
      main = paste("Prediction", predIdx)
    )

  }

  #-----------------

  # Standardized scale:

  panelCount = 1
  panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMargZ_intercept"))
  histInfo = plotPost(
    zbeta0,
    cex.lab = 1.75,
    showCurve = showCurve,
    xlab = bquote(z * beta[0]),
    main = "Intercept"
  )

  for (bIdx in 1:ncol(beta)) {
    panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMargZ_", bIdx))
    histInfo = plotPost(
      zbeta[, bIdx],
      cex.lab = 1.75,
      showCurve = showCurve,
      xlab = bquote(z * beta[.(bIdx)]),
      main = xName[bIdx]
    )
  }

  panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMargZ_scale"))
  histInfo = plotPost(
    zVar,
    cex.lab = 1.75,
    showCurve = showCurve,
    xlab = bquote(z * tau),
```

```
    main = paste("Scale")
  )

  panelCount = plot_hist(panelCount, saveName = paste0(saveName,
"PostMargZ_rsq"))
  histInfo = plotPost(
    Rsq,
    cex.lab = 1.75,
    showCurve = showCurve,
    xlab = bquote(R ^ 2),
    main = paste("R-squared")
  )

  panelCount = plot_hist(panelCount,
                         finished = TRUE,
                         saveName = paste0(saveName, "PostMargZ"))

}
```

4. Code for getting the data into R environment

```
crime_demo_dat <- read_csv("clinton1.csv")
```

5. Code for performing the Descriptive Check

```
# Kernel density estimation - dependent variable
ggplot(crime_demo_dat, aes(x = crime_index)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  ggtitle("\nKernel Density Plot for Crime Index\n") +
  xlab("Crime Index") +
  ylab("Density")

#-------------------

#Kernel density estimation - independent variables
kde_plot_1 <- ggplot(crime_demo_dat, aes(x = median_age)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Median Age") +
  ylab("Density")

kde_plot_2 <- ggplot(crime_demo_dat, aes(x = savings)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Savings") +
  ylab("Density")

kde_plot_3 <- ggplot(crime_demo_dat, aes(x = per_capita_income)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Per Capita Income") +
  ylab("Density")

kde_plot_4 <- ggplot(crime_demo_dat, aes(x = poverty_prcnt)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Poverty Percentage") +
  ylab("Density")

kde_plot_5 <- ggplot(crime_demo_dat, aes(x = veterans_prcnt)) +
```

```
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Veterans Percentage") +
  ylab("Density")

kde_plot_6 <- ggplot(crime_demo_dat, aes(x = population_density)) +
  geom_density(color = "darkblue", fill = "lightblue") +
  xlab("Population Density") +
  ylab("Density")

fig_kde <- ggarrange(
  kde_plot_1,
  kde_plot_2,
  kde_plot_3,
  kde_plot_4,
  kde_plot_5,
  kde_plot_6,
  nrow = 2,
  ncol = 3
)
fig_kde

#-------------------

# Scatter plots
scatter_plot_1 <- ggplot(crime_demo_dat, aes(x = median_age, y =
crime_index)) +
  geom_point() +
  xlab("Median Age") +
  ylab("Crime Index")

scatter_plot_2 <- ggplot(crime_demo_dat, aes(x = savings, y = crime_index))
+
  geom_point() +
  xlab("Savings") +
  ylab("Crime Index")

scatter_plot_3 <- ggplot(crime_demo_dat, aes(x = per_capita_income, y =
crime_index)) +
  geom_point() +
  xlab("Per Capita Income") +
  ylab("Crime Index")

scatter_plot_4 <- ggplot(crime_demo_dat, aes(x = poverty_prcnt, y =
crime_index)) +
  geom_point() +
  xlab("Poverty Percentage") +
  ylab("Crime Index")

scatter_plot_5 <- ggplot(crime_demo_dat, aes(x = veterans_prcnt, y =
crime_index)) +
  geom_point() +
  xlab("Veterans Percentage") +
  ylab("Crime Index")

scatter_plot_6 <- ggplot(crime_demo_dat, aes(x = population_density, y =
crime_index)) +
  geom_point() +
  xlab("Population Density") +
  ylab("Crime Index")
```

```
fig_scatter <- ggarrange(
  scatter_plot_1,
  scatter_plot_2,
  scatter_plot_3,
  scatter_plot_4,
  scatter_plot_5,
  scatter_plot_6,
  nrow = 2,
  ncol = 3
)
fig_scatter

#--------------------

#box plots
meltData <- melt(crime_demo_dat)

ggplot(meltData, aes(factor(variable), value)) +
  geom_boxplot(fill = "lightblue") +
  facet_wrap( ~ variable, scale = "free") +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())

#--------------------

#correlation between dependent variables
cormat <- round(cor(crime_demo_dat[, -7]), 2)

get_upper_tri <- function(cormat) {
  cormat[lower.tri(cormat)] <- NA
  return(cormat)
}
upper_tri <- get_upper_tri(cormat)

melted_cormat <- melt(cormat, na.rm = TRUE)

ggplot(data = melted_cormat, aes(Var2, Var1, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(
    low = "blue",
    high = "red",
    mid = "white",
    midpoint = 0,
    limit = c(-1, 1),
    space = "Lab",
    name = "Pearson\nCorrelation"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(
    angle = 45,
    vjust = 1,
    size = 12,
    hjust = 1
  )) +
  coord_fixed() +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
  theme(
    axis.title.x = element_blank(),
```

```
    axis.title.y = element_blank(),
    panel.grid.major = element_blank(),
    panel.border = element_blank(),
    panel.background = element_blank(),
    axis.ticks = element_blank())
```

6. Code for preparing the data for JAGS

```
#extract sample
set.seed(999)
crime_demo_dat_sample <- sample_n(crime_demo_dat,
                                  round(dim(crime_demo_dat)[1] * 0.1, 0))

#separate dependent and independent variables
y = as.matrix(crime_demo_dat[, "crime_index"])
x = as.matrix(crime_demo_dat[, -7])

#specify input values for dependent variables
xPred = array(NA, dim = c(6, 6))
xPred[1, ] = c(33, 87000, 50000, 7.02, 10, 200)
xPred[2, ] = c(18, 30000, 15000, 17.19, 19.22, 1)
xPred[3, ] = c(60, 690000, 4300, 50.89, 29, 35430)
xPred[4, ] = c(42, 50000, 26700, 3.6, 4.58, 8080.59)
xPred[5, ] = c(50, 400000, 16300, 34.77, 7.77, 3000)
xPred[6, ] = c(27, 10800, 12200, 5.2, 18.59, 131)

#wrap the data in list for JAGS
dataList <- list(
  x = x ,
  y = y ,
  xPred = xPred ,
  Nx = dim(x)[2] ,
  Ntotal = dim(x)[1]
)

# specify initial values for MCMC model
initsList <- list(zbeta0 = 100,
                  zbeta = c(100, 1, 1, 1, 1, 1),
                  Var = 1000)
```

7. Code for model specification

```
modelString = "

# Standardize the data:
data {
  y_sd <- sd(y)
  for (i in 1:Ntotal) {
    zy[i] <- y[i,] / y_sd
  }
  for (j in 1:Nx) {
    x_sd[j] <- sd(x[, j])
    for (i in 1:Ntotal) {
      zx[i, j] <- x[i, j] / x_sd[j]
    }
  }
}
```

```
#--------------------

# Specify the model for scaled data:
model {

  for (i in 1:Ntotal) {
    zy[i] ~ dgamma((mu[i]^2) / zVar, mu[i] / zVar)
    mu[i] <- zbeta0 + sum(zbeta[1:Nx] * zx[i, 1:Nx])
  }

  # Priors on standardized scale:
  zbeta0 ~ dnorm(0, 1 / (20^2))
  zbeta[1] ~ dnorm(-9 / x_sd[1], 1 / (2.7/x_sd[1]^2))
  zbeta[2] ~ dnorm(0 / x_sd[2], 1 / (2.7/x_sd[2]^2))
  zbeta[3] ~ dnorm(0.000001 / x_sd[3], 1 / (2.7/x_sd[3]^2))
  zbeta[4] ~ dnorm(0.8 / x_sd[4], 1 / (0.3/x_sd[4]^2))
  zbeta[5] ~ dnorm(-4 / x_sd[5], 1 / (15/x_sd[5]^2))
  zbeta[6] ~ dnorm(0.0001 / x_sd[6], 1 / (2.7/x_sd[6]^2))
  zVar ~ dgamma(0.01, 0.01)

  #------------------

  # Transform to original scale:
  beta[1:Nx] <- (zbeta[1:Nx] / x_sd[1:Nx]) * y_sd
  beta0 <- zbeta0 * y_sd
  tau <- zVar * (y_sd)^2

  #------------------

  # Compute predictions at every step of the MCMC
  for (i in 1:6){
    pred[i] <- beta0 + beta[1] * xPred[i, 1] + beta[2] * xPred[i, 2] +
beta[3] * xPred[i, 3] +
               beta[4] * xPred[i, 4] + beta[5] * xPred[i,5] + beta[6] *
xPred[i,6]
  }

}
"
```

8. Code for running the MCMC simulation

```
# Specify MCMC settings
adaptSteps = 1000
burnInSteps = 6000
nChains = 5
thinSteps = 135
numSavedSteps = 2100

# Run the model
start_time <- Sys.time()

runJagsOut <- run.jags(
  method = "parallel",
  model = modelString,
  monitor = c("beta0", "beta", "tau", "zbeta0", "zbeta", "zVar", "pred"),
  data = dataList,
  inits = initsList,
  n.chains = nChains,
```

```
  adapt = adaptSteps,
  burnin = burnInSteps,
  sample = numSavedSteps,
  thin = thinSteps,
  summarise = FALSE,
  plots = FALSE
)

codaSamples = as.mcmc.list(runJagsOut)

end_time <- Sys.time()
run_time <- end_time - start_time
print(run_time)

#save image
#save.image(file = "run5_p.RData")

#load required image
#load(file = "run4_p.RData")
```

9. Code for MCMC diagnostic checks

```
#MCMC check
diagMCMC(codaSamples, parName = "beta0")
diagMCMC(codaSamples, parName = "beta[1]")
diagMCMC(codaSamples, parName = "beta[2]")
diagMCMC(codaSamples, parName = "beta[3]")
diagMCMC(codaSamples, parName = "beta[4]")
diagMCMC(codaSamples, parName = "beta[5]")
diagMCMC(codaSamples, parName = "beta[6]")
diagMCMC(codaSamples, parName = "tau")

diagMCMC(codaSamples, parName = "pred[1]")
diagMCMC(codaSamples, parName = "pred[2]")
diagMCMC(codaSamples, parName = "pred[3]")
diagMCMC(codaSamples, parName = "pred[4]")
diagMCMC(codaSamples, parName = "pred[5]")
diagMCMC(codaSamples, parName = "pred[6]")

#summary
summary <- summary_MCMC(codaSamples = codaSamples)
summary
```

10. Code for displaying posterior distributions and kernel density estimate of observed versus predicted.

```
#specify comparison values
compVal <- data.frame(
  "beta0" = NA,
  "beta[1]" = NA,
  "beta[2]" = NA,
  "beta[3]" = NA,
  "beta[4]" =  NA,
  "beta[5]" =  NA,
  "beta[6]" =  NA,
  "tau" = NA,
```

```
  check.names = FALSE
)

#------------------

#plot posterior distributions
plot_MCMC(
  codaSamples = codaSamples,
  data = crime_demo_dat,
  xName = c(
    "median_age",
    "savings",
    "per_capita_income",
    "poverty_prcnt",
    "veterans_prcnt",
    "population_density"
  ),
  yName = "crime_index",
  compVal = compVal
)

#------------------

#compute required values
coefficients <- summary[c(2:8), 3]
Variance <- summary[9, 3]

meanGamma <-
  as.matrix(cbind(rep(1, nrow(x)), x)) %*% as.vector(coefficients)
randomData <- rgamma(n = length(y),
                     shape = meanGamma ^ 2 / Variance,
                     rate = meanGamma / Variance)

predicted <- data.frame(crime_index = randomData)
observed <- data.frame(elapsed = y)

predicted$type <- "Predicted"
observed$type <- "Observed"
dataPred <- rbind(predicted, observed)

#------------------

# Display the density plot of observed data and predicted
ggplot(dataPred,
       aes(crime_index, fill = type)) +
  geom_density(alpha = 0.2) +
  xlab("No. of Observations") +
  ylab("Density") +
  ggtitle("Density Estimate: Observed v/s Predicted") +
  theme(plot.title = element_text(hjust = 0.5))
```

(E.O.F.)