BAYESIAN LEARNING AND MONTECARLO SIMULATION

Project - Olympic dataset

Emelie Rosenberg
karinemelie.rosenberg@mail.polimi.it
10763429

July 8, 2020

1 Introduction

This report will cover the analysis of the London 2012 Olympic dataset. The dataset gives the information about the 203 countries that participated in the Olympic games in London 2012, the number of gold, silver, bronze and total medals medals won by each country, Borda points by country, income per capita (in \$ 10,000), population size (in 1,000,000,000), gross domestic product (GDP= income per capita multiplied by population size) and the polynomial variables of income per capita squared, population size squared, income per capita cubed, population size cubed, gross domestic product squared, gross domestic product cubed, natural log of income per capita, natural log of population size, and natural log of GDP. The response variable in this project is TotalMedals. [1],[2],[3].

The analysis will cover both frequentist and bayesian statistic approach and the goal is to find a good model and prediction for the data. The models and theory will be taken from the books *Bayesian Core - A Practical Approach*, 2007, Robert P. Christian and Marin Jean-Michel and *Applied Bayesian Statistics*, 2013, Cowles Mary Kathryn. If other sources is used it will be presented in the section sources.

2 Data overview

The whole code is to find through the link in the top of Appendix. Below in figure 1 the boxplot and correlation plot of the explanatory variables, in our case the data mentioned above in the introduction, are presented. We can see from the boxplot that we have a lot of outliers. The same applies to the respons variable TotalMedals as we can see in the histogram in figure 2. In the correlation plot we can see that we have high correlation between the different medals, TotalMedals and BordaPoints. This is expected consider the Totalmedals is the sum of gold, silver and bronze medals and the BordaPoints are dependent of the amount of medal a country gets. In figure 2 we can see that we have a lot of values around zero and like the explanatory values there's some outliers in the response variable as well. We now know some more about our dataset and that it probably will be quit hard to predict consider the concentration of data and the outliers.

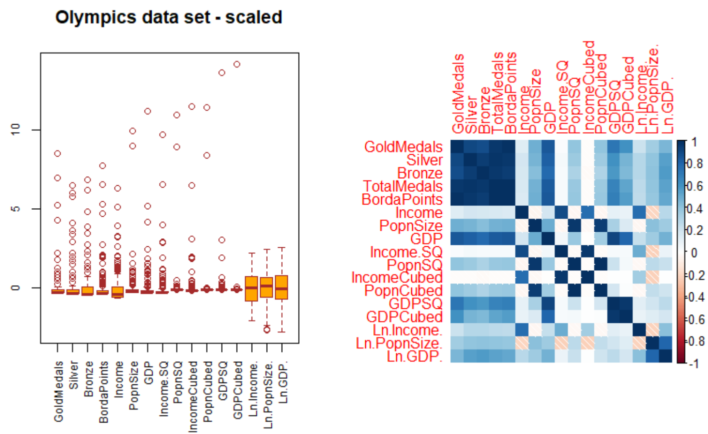

Figure 1. Boxplot and correlation plot over the features

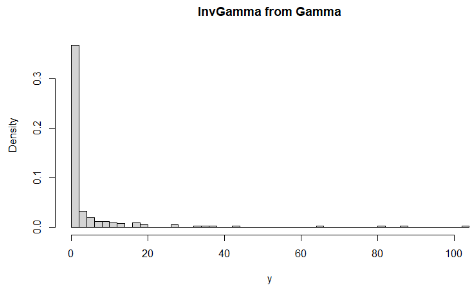

Figure 2. Histogram over the response TotalMedals

3 Model selection and Regression

In this chapter both modeling with conjugate models, model choice with help of BIC will be handled. In the first part a conjugate model is used to model our data. Later a BIC model will be used together with regression to find the ultimate regression model. I've chosen to work with all the covariates except the highly correlated ones, all the medals and bordapints. I have also chosen to exclude the ln tranformed ones because I'm working with Income, GDP and PopnSize.

We want to use the model for prediction later one which means that we can't choose the best model on the whole dataset, we have to divide it in two parts, one for training and one for testing. I've chosen to take away 40 countries for testing, 20% of the dataset, and use the rest for training. I random select which data I will use for training. During the modelling process I use `set.seed(1234)`.

3.1 Conjugate model

The response variable TotalMedals is only integers and are never under zero, therefore it's suitable to use a poisson-gamma conjugate model below.

$$x_i \sim Poi(\lambda) \tag{3.1}$$
$$\lambda \sim Ga(\alpha, \beta), \tag{3.2}$$

The likelihood is poisson distributed and the prior is the gamma. They are both presented in equation 3.2. The posterior then becomes a gamma distribution of the type: $Ga(\alpha_n, \beta_n)$.

$$\alpha_n = \alpha + \sum_{i=1}^{n} x_i \tag{3.3}$$
$$\beta_n = \beta + n \tag{3.4}$$

With the mean of the response variable we obtain the poisson likelihood, with the mean and variance from the response variable β and α for the prior was calculated according to the formulas from a gamma distribution. The values obtain was $\alpha = 0.124$ $\beta = 0.0262$

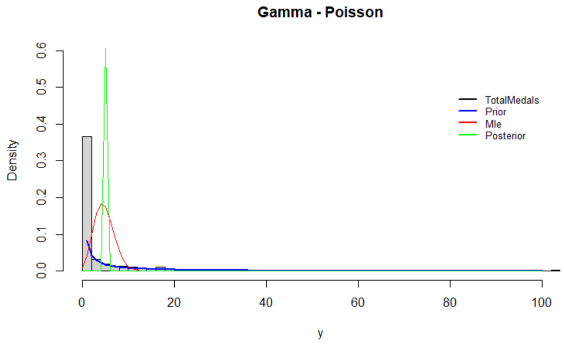

Figure 3. Histogram over the response TotalMedals with the Maximum likelihood estimation, posterior and prior distribution

It is possible to see that due to the high amount of zeros it's hard to fit the data so the conjugate model, poisson-gamma, isn't the best option since it doesn't take into account the high amount of zeros.

3.2 BIC - Bayesian Information Criterion

Instead of comparing the R^2 values to determine which model is the best we will use the BIC instead. The BIC is partly created to avoid overfitting as it has a penalty term for the number of parameters in the model. The definition of the BIC is presented in equation 3.5.

$$BIC = -2ln(likel\hat{i}hood) + (p+1)ln(n) \tag{3.5}$$

Where n is the number of observations in the model, and p is the number of predictors. To determine which model is the best we want to have the one that gives the lowest BIC-value. When n is high the BIC can be approximate to $BIC \approx -2ln(p(data|M))$ which means that we can choose the model with the highest log marginal likelihood due to the fact that it's negative proportional to the BIC. [4]

a) GLM and LM models

I choose to use both the `bas.glm` model which is a generalized linear model and `bas.lm` which is a linear model. Both models is using Bayesian adaptive sampling. I choose them both so I could compere the results in the end. For a summary over the top five models see Appendix.

```
cog.BIC.uni = bas.glm(y.train   ., data = olympic, family=poisson(), betaprior
= bic.prior(), modelprior = uniform())

cog.BIC = bas.lm(y.train   ., data = olympic, modelprior = uniform())
```

The model is with a uniform modelprior and the family poisson. For summary (the top five models) see Appendix C. To fit the best model we use it in our regression model.

```
cog.bestBIC.uni = bas.glm(y.train   ., data = olympic, family=poisson(), n.models
= 1, betaprior = bic.prior(), bestmodel = bestgamma.uni, modelprior = uniform())
```

```
cog.bestBIC = bas.lm(y.train   ., data = olympic, n.models = 1, bestmodel
= bestgamma, modelprior = uniform())
```

A summary of the results from the best BIC regression models is showed in the tables below. In the tables the posterior mean and standard deviation together with the lower and upper quantile is showed. The best model for the glm was the model with all the covariates except IncomeCubed. For the lm the best model had the covariates. We can see in the table below that PopnSize and PopnCubed has a big influence on the data because it's weight (βs) are bigger (in absolute value) then the rest.

	post mean	**post sd**	**2.5%**	**97.5%**
Intercept	0.7458567	0.06151263	0.6243577	0.8673558
Income	0.9228443	0.20248436	0.5228993	1.3227892
PopnSize	1.8953225	0.25906463	1.3836208	2.4070241
GDP	1.8066629	0.16100243	1.4886526	2.1246731
Income.SQ	-1.0972988	0.29113129	-1.6723382	-0.5222595
PopnSQ	-8.2341109	1.02265289	-10.2540438	-6.2141780
IncomeCubed	0.0000000	0.00000000	0.0000000	0.0000000
PopnCubed	6.6348603	0.82452073	5.0062759	8.2634447
GDPSQ	-3.3478497	0.29113965	-3.9229055	-2.7727938
GDPCubed	2.0012307	0.18961743	1.6267004	2.3757610

Table 1. BAS.glm model

The best model for the lm was the model with the covariates PopnSize, GDP, PopnSQ, PopnCubed, GDPSQ and GDPCubed. We can see in the table below that GDP, PopnCubed and GDPSQ has a big influence on the data because it's weight (βs) are bigger (in absolute value) then the rest.

	post mean	**post sd**	**2.5%**	**97.5%**
Intercept	5.114458	0.4712275	4.184045	6.044871
Income	0.000000	0.0000000	0.000000	0.000000
PopnSize	8.299289	3.0515083	2.274251	14.324326
GDP	23.294755	2.3329548	18.688463	27.901047
Income.SQ	0.000000	0.0000000	0.000000	0.000000
PopnSQ	-81.744345	15.1316912	-111.621047	-51.867644
IncomeCubed	0.000000	0.0000000	0.000000	0.000000
PopnCubed	75.974944	13.3883035	49.540467	102.409422
GDPSQ	-31.304198	6.0855528	-43.319791	-19.288605
GDPCubed	20.017176	4.5433262	11.046626	28.987727

Table 2. BAS.lm model

3.3 Zellner's prior

This model is more of a long shot. It uses a Gaussian prior for β and a improper prior for σ^2. Considering that we have data that's more a poisson this probably won't work as good as the other models. But I would like to try one of the models that created for dataset were you don't ave much information about the prior. I've choosen the Zellner's G-prior since it only require tuning of two variables, c and $\tilde{\beta}$. The β estimate is given by equation 3.6 and c is a constant that can be interpreted as a measure of the amount of information available in the prior relative to the sample.

$$E^{\pi}[\beta|\mathbf{y},X] = \frac{1}{c+1}(\tilde{\beta} + c\hat{\beta}) \tag{3.6}$$

The first values of $\hat{\beta}$ is a maximum likelihood estimate solved by the least squares problem:

$$\hat{\beta} = (X^T X)^{-1} X^T \mathbf{y} \tag{3.7}$$

The $\hat{\beta}$ is then updated by the Zellner's prior estimate of beta shown above. By using the new βs we can calculate the new y that later can be used for prediction.

$$y_{pred} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n \tag{3.8}$$

n in our case goes from 0 to 9. The values of the new βs is:

explanatory	β
Intercept	3.796551e-01
Income	-1.225634e-01
PopnSize	4.958643e+01
GDP	1.951071e+02
Income.SQ	-1.070410e-01
PopnSQ	-4.924783e+02
IncomeCubed	6.880633e-03
PopnCubed	3.617159e+02
GDPSQ	-2.106913e+02
GDPCubed	2.581607e+01

4 JAGS

Just Another Gibbs Sampler (JAGS) is a program used for Bayesian modelling using Markov chain Monte Carlo. To find a model that can handle the high amount o zeros a Zero Inflated Poisson model seems like a good idea. The ZIP-model has two parts, one that is a logit model that handles the zeros and one that are a poisson model that handle the non-zero elements (this is a simplified explanation, both parts of the ZIP can generate zeros!). The two parts of the model is described as:

$$Pr(Y=0) = \pi + (1-\pi)e^{-\lambda} \tag{4.1}$$

$$Pr(Y=y_i) = (1-\pi)\frac{\lambda^{y_i} e^{-\lambda}}{y_i!} \tag{4.2}$$

The model that was used for the simple JAGS model is the following one:

```
ZIPois_string <- "model{
  # likelihood
    for (i in 1:n) {
    y[i]~dpois(mu[i])
    mu[i] <- lambda*z[i] + 0.00001  ## 0 is not admitted for poisson
    z[i]~dbern(1-q)
    }
    #predictive
    yp~dpois(mup)
    mup<-lambda*zp+ 0.00001
```

```
    zp~dbern(1-q)
    # prior
    lambda ~ dgamma(200,200)
    q ~ dbeta(50,1)
}"
```

From that model we obtain the following plot:

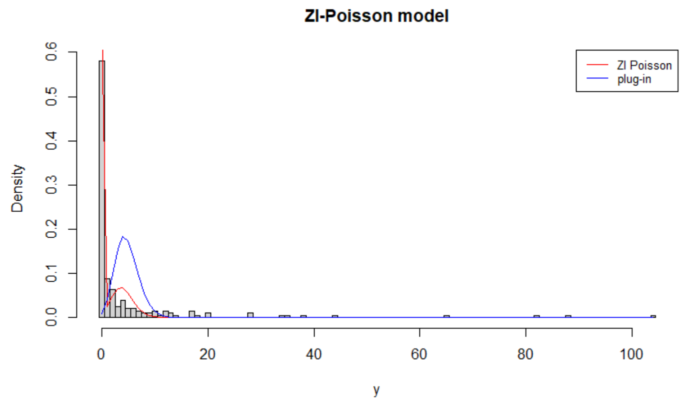

Figure 4. Histogram over the response TotalMedals with the Zero Inflated Poisson model

We can see in the plot 4 that it looks like a much better fit then the conjugate model. For plots over the trace and density see Appendix D, in those plots it's possible to see that the are some fluctuation around λ. We also tried to do a more complex model to update the βs and use it for prediction but the model never converge. We tried to change the different parameters to find converges but it failed. One of the reason that it didn't converge can be the for the complex number of covariates and the lack of information we have about the prior.

5 Prediction

For each model prediction will be made with the test dataset. The result will be measured with Mean Square Error (MSE) and we'll see which model that perform the best.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{5.1}$$

I also used the build in model with the command `bayesglm(TotalMedals  ., data = olympic.train, family=poisson)`. Summary over the bayesglm is find in Appendix E, in that summary it's possible to see that all the covariates except IncomeCubed are significant at level 0.01 or higher. For that model the following predictor was used:

```
pred.bay.glm = predict.glm(baymod, newdata = olympic.test, type = "link")
```

For Zellner's prior I used the equation 3.8, for the BAS.glm and BAS.lm we used the following Bayesian predictor:

```
pred = predict(cog.bestBIC, newdata = olympic.test, top=1)
```

For the summary of the predictors see Appendix E. In the table below you can see the different MSE for the prediction for the different models.

Model	MSE
Bayesglm	103.432
Zellner's	160.4
bas.glm	103.414
bas.lm	32.03

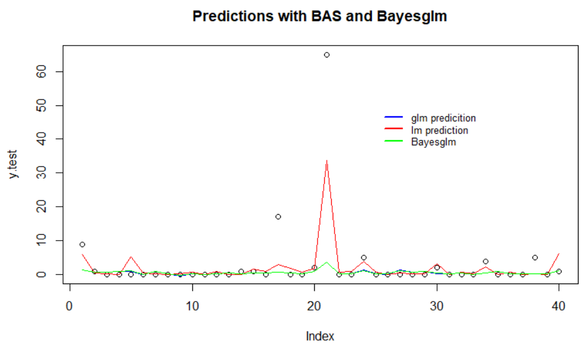

Figure 5. Prediction with BAS.glm, BAS.lm and Bayesglm togheter with real data

In the plot 5 we can see that the predictions from Bayesglm and BAS glm are almost completely identical, which makes sense consider how similar they are in the MSE.

6 Conclusion

- For next time it would be better to divide the data into three sets, train, test and validation data for the robustness in the model selection.
- The BAS linear regression model performed much better then the rest. It didn't use any of the covariates with Income which tells us that the response variable doesn't depend as much on Income compere to the other. We can see that it performs better both in the MSE and in the plot 5.
- In the Bayesglm model IncomeCubed was the only covariate that wasn't significant, Income and Income.SQ had low values which means they aren't as important as the rest. Like the linear model this model also shown that Income is less important then the rest for modeling the response variable.
- The Zellner's prior that I used didn't work well at all for the model, it can be because it uses a Gaussian prior for β.
- It seems like the linear model is much better to fit this data with the high amount of zeros compere to the glm models. If it wasn't for the high amount of zeros the glm model with family poisson most likely would have been better. Using JAGS with zero inflated poisson is probably the best option for this dataset. JAGS are powerful and zero inflated poisson is made for data with a lot of zeros.

7 Sources

[1] http://espn.go.com/olympics/summer2012/medals
[2] List of nations at http://www.london2012.com/countries/
[3] The data on income per capita and population size by country http://www.csuchico.edu/math/thesesprojects.s
[4] BIC https://statswithr.github.io/book/bayesian-model-selection.html

8 Appendix

8.1 A

The whole dataset and the code will be able to find at the GitHub site:

Code for the project:
https://github.com/rooosenberg/BayStat_Project

8.2 B

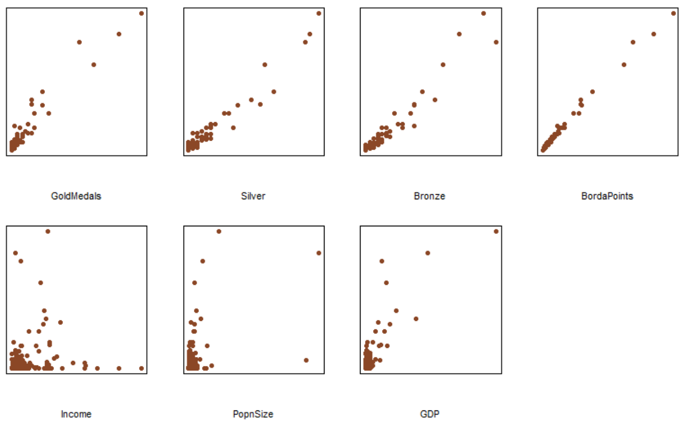

Figure 6. The different explanatory variables in relationship with TotalMedals

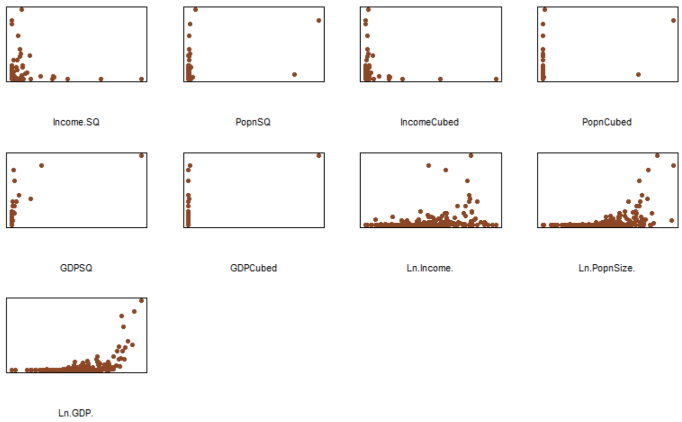

Figure 7. The different explanatory variables in relationship with TotalMedals

8.3 C

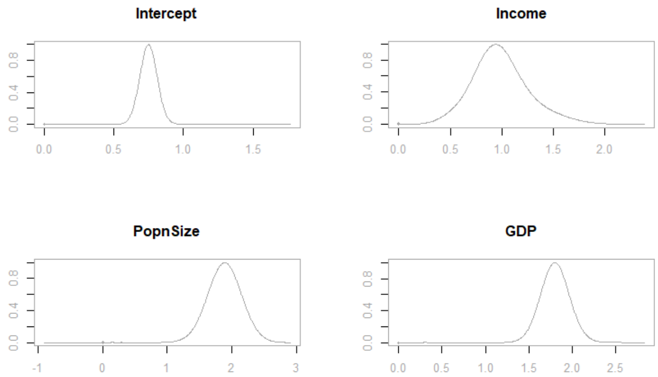

Figure 8. The different beta for the different explanatory variables

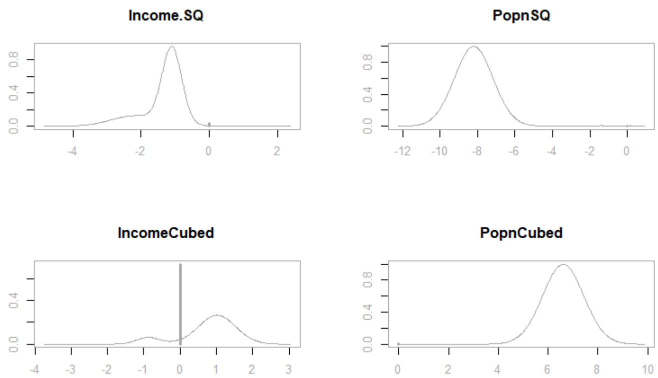

Figure 9. The different beta for the different explanatory variables

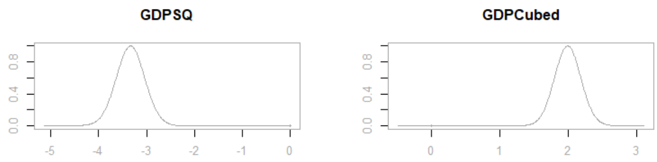

Figure 10. The different beta for the different explanatory variables

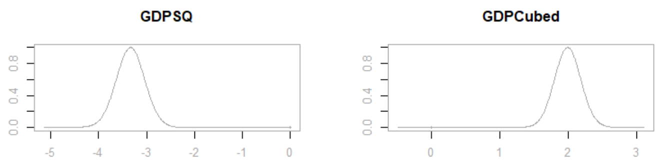

Figure 11. The different beta for the different explanatory variables

Summary over the topfive models for glm and lm with BAS

	P(B != 0 | Y)	model 1	model 2	model 3	model 4	model 5
Intercept	1.0000000	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
Income	0.2276442	0.0000	0.0000000	1.0000000	1.0000000	0.0000000
PopnSize	0.6708004	1.0000	0.0000000	1.0000000	0.0000000	1.0000000
GDP	1.0000000	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
Income.SQ	0.1509686	0.0000	0.0000000	0.0000000	0.0000000	1.0000000
PopnSQ	0.9961598	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
IncomeCubed	0.1419135	0.0000	0.0000000	0.0000000	0.0000000	0.0000000
PopnCubed	0.9961970	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
GDPSQ	0.9992958	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
GDPCubed	0.9945802	1.0000	1.0000000	1.0000000	1.0000000	1.0000000
BF	NA	1.0000	0.3169609	0.1687922	0.1509864	0.1097783
PostProbs	NA	0.4552	0.1443000	0.0768000	0.0687000	0.0500000
R2	NA	0.8176	0.8090000	0.8190000	0.8131000	0.8180000
dim	NA	7.0000	6.0000000	8.0000000	7.0000000	8.0000000
logmarg	NA	121.5928	120.4438704	119.8137603	119.7022817	119.3835551

Figure 12. Top five models lm

```
> summary(cog.BIC.uni)
              P(B != 0 | Y)   model 1     model 2     model 3      model 4      model 5
Intercept            1.0000    1.0000    1.000000   1.0000000 1.000000e+00 1.000000e+00
Income               0.9919    1.0000    1.000000   1.0000000 0.000000e+00 0.000000e+00
PopnSize             0.9974    1.0000    1.000000   1.0000000 1.000000e+00 0.000000e+00
GDP                  0.9991    1.0000    1.000000   1.0000000 1.000000e+00 1.000000e+00
Income.SQ            0.9608    1.0000    1.000000   0.0000000 0.000000e+00 1.000000e+00
PopnSQ               0.9971    1.0000    1.000000   1.0000000 1.000000e+00 0.000000e+00
IncomeCubed          0.2662    0.0000    1.000000   1.0000000 0.000000e+00 0.000000e+00
PopnCubed            0.9960    1.0000    1.000000   1.0000000 1.000000e+00 0.000000e+00
GDPSQ                0.9983    1.0000    1.000000   1.0000000 1.000000e+00 1.000000e+00
GDPCubed             0.9985    1.0000    1.000000   1.0000000 1.000000e+00 1.000000e+00
BF                       NA    1.0000    0.319724   0.0544692 4.951446e-04 2.606693e-29
PostProbs                NA    0.7274    0.231400   0.0329000 2.900000e-03 1.000000e-03
R2                       NA    0.7126    0.713600   0.7104000 7.033000e-01 6.570000e-01
dim                      NA    9.0000   10.000000   9.0000000 7.000000e+00 5.000000e+00
logmarg                  NA -542.6884 -543.828735 -545.5985580 -5.502991e+02 -6.085053e+02
```

Figure 13. Top five models glm

8.4 D

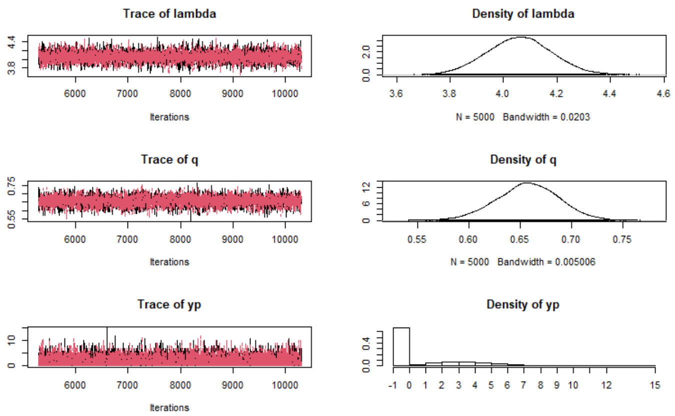

Figure 14. Data over the simple ZIpoisson JAGS model

Jags model for find beta

```
ZIPois_string <- "model{
  # likelihood
    for (i in 1:n) {
    y[i]~dpois(mu[i])
    mu[i] <- lambda[i]*z[i] + 0.00001  ## 0 is not admitted for poisson
    z[i]~dbern(1-q)
    lambda[i]=inprod(beta[],X[i,]) ##For every covariant

    }
    #predictive
    yp~dpois(mup)
    mup<-lambdap*zp+ 0.00001
    zp~dbern(1-q)

    # prior
    lambdap ~ dgamma(200,200)
    q ~ dbeta(50,1)

    # prior distributions
    beta[1:P] ~ dmnorm( mu.beta[], prior.T[,] )
```

```
    tau     ~ dgamma(50,1 )

    # prior
    c2 <- n

    # prior means
    for (j in 1:P){ mu.beta[j] <- 0.0 }

    # calculation of xtx
    for (i in 1:P){ for (j in 1:P){
       inverse.V[i,j] <- inprod( X[,i] , X[,j] )}}

    for(i in 1:P){ for (j in 1:P){
      prior.T[i,j] <- inverse.V[i,j] * tau /c2  }}
}"
```

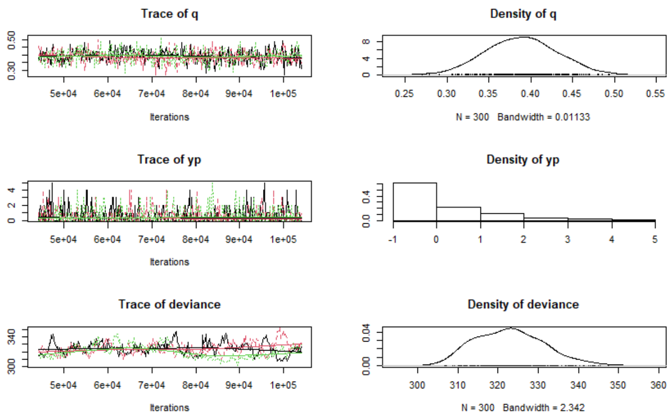

Figure 15. ZIpoisson JAGS model that failed to converge

8.5 E

For the Bayesglm, summary over the prediction

```
       Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.2534  0.2393  0.4365  0.6095  0.9136  3.7228
```

For the BAS.glm and BAS.lm is the same, summary over the prediction

```
    > summary(pred.uni)
              Length Class  Mode
fit           40     -none- numeric
Ybma          40     -none- numeric
Ypred         40     -none- numeric
postprobs      1     -none- numeric
se.fit         0     -none- NULL
se.pred        0     -none- NULL
se.bma.fit     0     -none- NULL
se.bma.pred    0     -none- NULL
```

```
df            1      -none- numeric
best          1      -none- numeric
bestmodel     1      -none- list
best.vars    10      -none- character
estimator     1      -none- character
```

Summary over bayesglm model

```
> summary(baymod)

Call:
bayesglm(formula = TotalMedals ~ ., family = poisson, data = olympic.train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.1631  -1.6675  -1.5097  -0.1723   7.9699

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.023709   0.106962   0.222  0.82458
Income        0.414664   0.096379   4.302 1.69e-05 ***
PopnSize     13.297777   1.859042   7.153 8.49e-13 ***
GDP          13.589190   1.163704  11.678  < 2e-16 ***
Income.SQ    -0.052520   0.019098  -2.750  0.00596 **
PopnSQ      -46.058372   5.657322  -8.141 3.91e-16 ***
IncomeCubed   0.001209   0.001090   1.109  0.26746
PopnCubed    28.747357   3.525694   8.154 3.53e-16 ***
GDPSQ       -19.406130   1.664312 -11.660  < 2e-16 ***
GDPCubed      2.221411   0.210614  10.547  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 2737.39  on 165  degrees of freedom
Residual deviance:  785.41  on 156  degrees of freedom
AIC: 1063

Number of Fisher Scoring iterations: 13
```