KU Leuven

Statistical Modelling: Final Assignment

Akshat Dwivedi (Student Number: )

Instructor: Prof. Gerda Claeskens

June 1, 2016

1 Part A

The dataset used in this part of the project contains information about Belgian municipalities for the year 2014. Our aim is to study how and whether these variables help explain the number of births in the municipalities. The variables in the dataset are: province, income, zero_income (individuals that do not pay taxes), population density, num. of marriages, num of divorces, population on the first and last day of the year, deaths, incoming and outgoing internal as well as international migration, number of nationality changes and the response variable which is number of births per 1000 (also known as crude birth rate (CBR)) inhabitants.

1.1 A.1

The average number of births per 1000 inhabitants and the average number of international migrants per province is given in table 1.

Province	Antwerp	Flemish Brabant	West Flanders	East Flanders	Hainaut	Liege	Limburg	Luxembourg	Namur	Walloon Brabant	Brussels
Num. of Incoming International Migrants	2575	1263	1502	1791	1875	3265	3425	2200	2854	2872	4020
births per 1000	13.77	12.82	12.20	12.46	12.39	14.61	15.44	14.08	15.82	13.83	20.17

Table 1: Section A.1: Average number of incoming international migrants and births per 1000 (crude birth rate) for each province in Belgium.

We wish to test whether the provinces Flemish Brabant and Brussels have similar number of incoming international migrants and births per 1000. For Brussels, we have 3 observations in our dataset and for Flemish Brabant, we have 13 observations. Since 13 is quite small and 3 is too small, a t-test is out of the question. It would be unreasonable to assume that the distributions are normal with such a low sample size. We can perhaps use a nonparametric test such as Mann-Whitney U test (for 2 samples) or Kruskal-Wallis (extension of Mann-Whitney for more than 2 groups), however, either way, 3 observations in one group is going to be problematic for drawing inferences. This small number is due to the seed chosen for obtaining a subset of the full dataset for analysis. We present the results from these tests below but these are not desirable since the CIs will be too wide. For this, we used the Mann-Whitney test which is a two-sample test for difference in location with $H_0 : \mu_1 - \mu_2 = 0$ vs $H_a : \mu_1 - \mu_2 \neq 0$. The sample sizes for each of the groups is $n_1 = 13$ (Flemish Brabant) and $n_2 = 3$ (Brussels).

For the variable births per 1000, the estimated difference in location is -9.0207 (95% CI: [-14.134, 0.359]) with a $p-$value of 0.05714. The effect size reported here is negative, which means that the average birth rate for Flemish Brabant is lower than than the average birth rate for Brussels although this difference is not statistically significant. On the other hand, for the number of incoming international migrants, the estimated dif-

ference in location is -2582 (95% CI: [-7134, 431]) with a $p-$value of 0.05714. The effect size is negative which means the average number of international incoming migrants is lower in Flemish Brabant compared to Brussels (as is expected), however, this difference is not statistically significant. The estimated CIs are really wide, most likely due to a small number of sample sizes for each of the groups and hence, the conclusions about the differences in group means are not particularly precise.

As an aside, we tried this on the full dataset with $n_{vlaams} = 65$ and $n_{BRU} = 19$ and this was enough to narrow the CIs and it was observed that the conclusions drawn above are the same (birth rate ES: -2.951, 95% CI [-7.61,-1.34], $p = 0.002$; international migration ES: -1609, 95% CI [-2328, -290], $p = 0.016$; $p-$values are two sided).

1.2 A.2

In this part we fit a flexible regression model $log(BirthPer1000) = f(Intl\_Mig) + \varepsilon$, $\varepsilon_i \sim$ Poisson(BirthPer1000_i) (where BirthPer1000$_i$ = $exp(\beta$Intl_Mig$_i))$ is the link function for the Poisson model) using penalized smoothing splines. The fitted model using splines has ≈ 5 degrees of freedom ($df = 5.456$). Inspecting the fitted curve, we pick (50, 400, 1300, 9500) as the four knots for the linear spline basis function. The fitted spline and the parametric line (GLM with Gamma errors and log link) is given in figure 1.

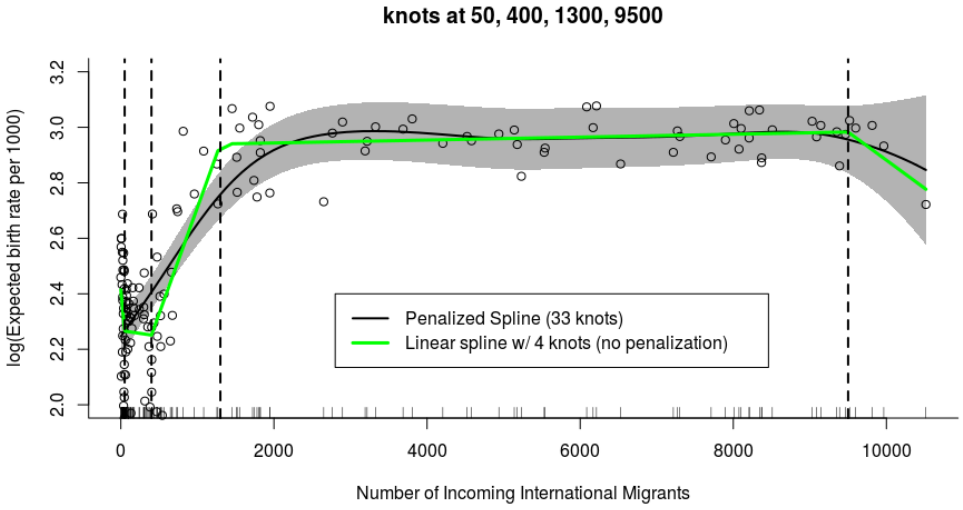

Figure 1: Section A.2: Fitted spline model with birth rate per 1000 as response and International Incoming Migrants as predictor. The black line is the spline fit; the green line is from the parametric (piecewise linear) fit with knots at (50,400,1300,9500); the black dashed lines indicate the positions of the knots. The model does not seem to fit well on $x \in [0, 500]$ as most of the observed values are far from either of the fitted values.

We test which knots to keep in the parametric model using 1) likelihood ratio tests and 2) information criteria. Based on the model summary, we use the likelihood ratio test between the model with these 4 knots and the model with knots only at 400 and 1300. The resulting $\chi^2 = 4.825$ with $df = 2$ and the corresponding p-value is 0.089. This shows that the likelihood does not differ much between the two models. The BIC for model with 2 knots is 699.42 vs 694.22 for the model with 4 knots. Based on these, it is clear that the knot at 9500 can be removed, however, it is not entirely evident whether the knot at 50 should be removed or not. We decide to remove the knot at 50 and only keep the knots at 400 and 1300, which results in a piecewise linear model. The equation for the chosen model is

$$\log(y_i) = \beta_0 + \beta_1 x_{1i} + b_1(x_{1i} - 400)_+ + b_2(x_{1i} - 1300)_+ + \varepsilon_i$$

with $\varepsilon_i \sim \Gamma(\theta, k)$ and $\mathrm{y}_i = \mathrm{birthRatePer1000}_i$ and x_1 is number of incoming international migrants.

1.3 A.3

In this section we fit an additive model with Income (in 1000s euros) and Zero Income (number of people that did not pay tax) to the response births per 1000 people (crude birth rate) using penalized smoothing splines using the spm function from the R package SemiPar. The equation for this model is

$$\log(\mathrm{birthsPer1000}_i) = \beta_0 + f(\mathrm{Income}_i) + g(\mathrm{ZeroIncome}_i) + \varepsilon_i$$

where $\varepsilon_i \sim \mathrm{Poisson}(\lambda_i)$; $\lambda_i = \mathrm{birthsPer1000}_i$.

The output from the fitted model shows that the estimated degrees of freedom (df) for each of the predictors is approximately 1, which implies that a (parametric) linear term adequately captures the trend in the model and that smoothing splines may not be necessary and a parametric approach will suffice. Figure 2 shows the fitted spline to the the response variable. The x-axis is the corresponding predictor in the model and the y-axis is our response variable log(crude birth rate). The trend is linear, which means that a change in the levels of the predictors leads to a linear change in the levels of the response. Furthermore, from the plots we see that as a family's income increases, they tend to have lesser children on average.

Similarly, the second plot shows that families with smaller incomes that are exempt from paying taxes have a slightly larger log(crude birth rate). However, the wide confidence interval (shaded area) in the second right half of the plot indicates the larger uncertainty in this part of the graph, that is there are a much smaller number of regions with such a high level of individuals earning low wages. For Zero Income values between 0 and 100, there does seem to be a positive slope to the trend line which provides some evidence of lower income households having on average, a larger number of kids. It should be noted that here we have not taken the hierarchical nature of the data into account and fit the

model for Belgium as a whole (marginal model) instead of building a model with region as a random effect.

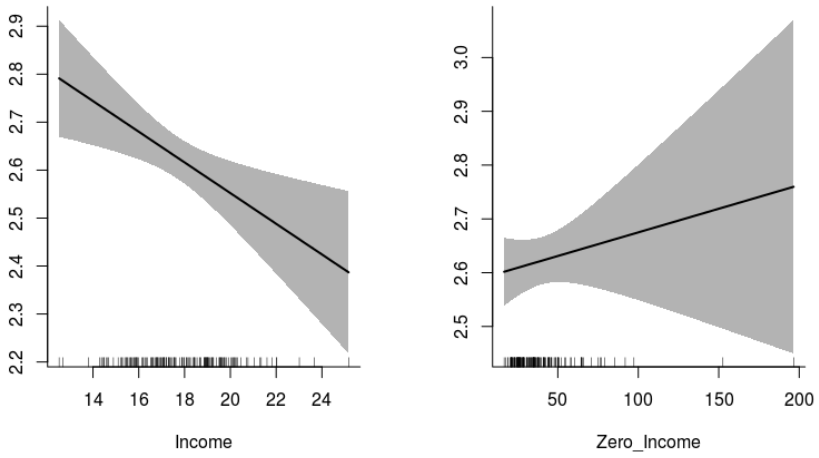

Figure 2: Section A.3: Plots of the fitted regression lines using semiparametric regression with Income and Zero Income as predictors. We can see that these predictors have a linear effect on the response variable crude birth rate (CBR) and hence splines are not necessary for these predictors and we can simply fit a parametric model to these predictors.

1.4 A.4

As demonstrated in the previous section, a parametric model is sufficient for studying the relationship between Income, Zero Income and the response variable births per 1000. In this section we fit a Gamma GLM with the log link and wish to test whether the main effects model is sufficient and that adding interaction or higher order terms are not necessary at the 1% level (which is taken to be the α for hypothesis testing). The reason for fitting the Gamma model in this case compared to Poisson for the previous case is due to lack of support for the Gamma family in the SemiPar R package. In the nonparametric case, we stick to the Poisson approximation whereas the GLM function allows us to fit a larger family from the exponential family of distributions. We fit two models in this section. The first model contains the main effects and has the following equation

$$\log(\text{birthsPer1000}_i) = \beta_0 + \beta_1 \text{Income}_i + \beta_2 \text{ZeroIncome}_i + \varepsilon_i$$

where $\varepsilon_i \sim \text{Gamma}(k, \theta)$; (k: shape, θ: scale) and the second model contains all second order terms. The equation for this is given below.

$$\log(\text{birthsPer1000}_i) = \beta_0 + \beta_1 \text{Income}_i + \beta_2 \text{ZeroIncome}_i + \beta_3 \text{Income}_i^2 + \beta_4 \text{ZeroIncome}_i^2 + \beta_5 \text{Income}_i * \text{ZeroIncome}_i + \varepsilon_i$$

where $\varepsilon_i \sim \text{Gamma}(k, \theta)$; (k: shape, θ: scale). For testing, we perform a likelihood ratio test between model 1 and model 2 since model 1 is nested in model 2. The null hypothesis H_0: model 1 is adequate vs the H_a: model 2 is adequate. The $p-$value for this is 0.32 (with $df = 3$) which means that we cannot reject the null hypothesis at the 1% level that model 2 is "better" than model 1, i.e., the data are more likely under model 2 than model 1. Thus, we can conclude that the data are almost equally likely under both models and hence, the additive model is sufficient for our purposes.

We also used the drop1 function in R which performs term-wise deletion and performs the likelihood ratio test between the models with the term included vs the model without the term (but all other terms in the model). This led to the same conclusion as the previous procedures in that dropping any of the second order terms does not change the log-likelihood by a significant amount and the models without the term are "as good as" the models with the term.

1.5 A.5

For this part we first fit a semiparametric model using spm in R package SemiPar. Income and Zero Income are included as parametric terms (of degree 1 as was shown in part A.3) and the rest of the terms are included in the non-parametric part. We dropped two predictors: population in January (which is highly correlated with population in December) and Nationality Out (countries to which people emigrated to) which is correlated with Nationality In. This was causing errors at the model fitting stage. Further, one could make the argument that we might be more concerned with nationals coming into Belgium than those going out of Belgium. We fit the model using the rest of the variables. Further, we include province as a random effect to account for the variability in different provinces.

Examining the fitted model, we see that the estimated degrees of freedom for all predictors except number of incoming international migrants is approximately 1 and for this variable is approximately 2.5. This means that a linear predictor for each of these predictors and a quadratic or cubic term for international incoming migrants may be sufficient and we can instead fit a parametric model instead of a semiparamteric one. We fit an initial model with all the predictors fit in the semiparametric model and include a quadratic as well as cubic term for number of incoming international migrants, and include province as a factor variable to potentially capture the mean shift in different provinces. Upon examining the coefficients, we narrow the set of predictors down to Income, Deaths, number of incoming and outgoing internal migrants, number of outgoing international migrants and the population in December. We fit a Gamma regression

model in both cases with the log link function. A likelihood ratio test between these two models gives a $\chi^2 = 18.359$ with $df_{full} - df_{reduced} = 15$. The resulting $p-$value is 0.2443 so we decide to stick with the reduced model.

Another interesting comparison is whether the parametric model with knots that was fit in section A.2 can provide a better fit than this reduced model with quadratic and cubic effects for international incoming migrants. Since these models are non-nested, we can compare these two models based on their respective BIC values. the BIC for the reduced model is 726.84 and for the model with knots (at 400 and 1300) is 698.21. Based on this, we choose the model with the knots. The coefficients are given in table 2.

Variables	Estimate	Std. Error	t value	Pr(> |t|)
(Intercept)	2.6275	0.1216	21.60	0.0000
Income	-0.0174	0.0066	-2.65	0.0089
Deaths	-0.0012	0.0003	-3.78	0.0002
IntMig_In	-0.0002	0.0001	-1.48	0.1398
IntMig_Out	0.0002	0.0001	1.70	0.0918
IntlMig_In	-0.0001	0.0001	-0.77	0.4431
pmax((IntlMig_In - 400), 0)	0.0008	0.0002	4.87	0.0000
pmax((IntlMig_In - 1300), 0)	-0.0007	0.0001	-11.16	0.0000
IntlMig_Out	-0.0004	0.0001	-3.11	0.0023
Population_Dec	0.0000	0.0000	3.35	0.0010

Table 2: Section A.5: Model summary for the final model in section A.5. This is a parametric model with piecewise linear terms for international incoming migrants by putting knots at (400, 1300) to capture the curvature in that variable.

The interpretation for the coefficients is straightforward, however some of the coefficients appear to have a very small impact on the response variable. Income, as noted in one of the previous sections has a negative impact on the crude birth rate, i.e. families with higher income on average have lesser kids. Regions with a higher number of deaths appear to have a negative impact n the birth rates as well, possibly due to that region containing (on average) a higher number of elderly people. The slope for the incoming international migrants (between 0 and 400) is positive which could be due to regions with a smaller number of international migrants (more rural than urban) which would explain the slightly higher birth rate. For a region with a moderate amount of incoming international migrants (400-1300), the birthrates seem to decrease with an increase in the number of migrants. The number of outgoing international migrants seems to have a small negative impact on the birth rate as well.

2 Part B

B.1: TIC for Poisson Regression

In this section, we derive the formula for Takeuchi's information criterion for Poisson regression using the canonical link function: $Y_i \sim \text{Poisson}(\mu_i)$ and $\mu_i = \exp(x_i^t\beta) \Leftrightarrow \log(\mu_i) = x_i^t\beta$. The log-likelihood of β is given as:

$$l(\beta) = \log\mathcal{L}(\beta) = \sum_{i=1}^{n} y_i \log(\mu_i) - \mu_i - \log(y_i) = \sum_{i=1}^{n} y_i(x_i^t\beta) - e^{(x_i^t\beta)} - \log(y_i) \tag{2.1}$$

Taking the derivative w.r.t. β, we have

$$\frac{\partial l(\beta)}{\partial(\beta)} = \sum_{i=1}^{n} y_i x_i - e^{(x_i^t\beta)} x_i = \sum_{i=1}^{n} \left(y_i - e^{(x_i^t\beta)}\right) x_i \tag{2.2}$$

which is the score equation (when equated with 0). Taking the second derivative of the log-likelihood w.r.t. β we get

$$\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^t} = \sum_{i=1}^{n} -e^{(x_i^t\beta)} x_i x_i^t \tag{2.3}$$

The equation for TIC is similar to AIC/BIC: $\text{TIC}(M) = -2\text{log-likelihood}_{\max}(M) + 2\text{Tr}(\hat{J}_n^{-1}\hat{K}_n)$ where $\hat{J}_n$ and $\hat{K}_n$ are given below by using the final terms from the derived equations (2.2) and (2.3) above

$$\hat{J}_n = -\mathbb{E}\left[-\exp(x_i^t\beta)x_i x_i^t\right] \quad \text{and} \quad \hat{K}_n = \mathbb{E}\left[\left((y_i - e^{(x_i^t\beta)})x_i\right)\left((y_i - e^{(x_i^t\beta)})x_i\right)^t\right] \tag{2.4}$$

B.2: Simulation Study

This part involves conducting a simulation study that compares AIC, BIC and TIC for Poisson regression. Five variables, $x_{i,j}$, $j = 2, \ldots, 6$, $i = 1, \ldots, n$ were sampled from Uniform(-1,1). The response variable Y was sampled from Poisson(μ_i) with

$$\mu_i = \exp\left(\frac{1}{3} + \frac{2}{3}x_{i,2} + 1 * x_{i,3} + \frac{4}{3}x_{i,4} + 0 * x_{i,5} + 0 * x_{i,6}\right)$$

The response vector is n x 1 and the design matrix is an n x 6 matrix with the first column containing the intercept (all 1's) and the rest of the columns containing the x_j's. Furthermore, we construct models with all the possible combinations of the covariates (which for 5 covariates results in $2^5 = 32$ models) and add the intercept term to each of the models. This simulation is run 1000 times for two sample sizes: $n = 50$ and $n = 400$ (table 3). The aim is to see which models are selected and with what frequency by AIC, BIC and TIC (i.e. picking a model with the smallest (A/B/T)IC value). The results are presented below.

Model	AIC	BIC	TIC
Only $\beta_1, \beta_2, \beta_3, \beta_4$	697	872	523
At least one of $\beta_1, \beta_2, \beta_3, \beta_4 = 0$	15	49	11
At least one of $\beta_5, \beta_6 \neq 0$	288	79	466

(a) $n = 50$

Model	AIC	BIC	TIC
Only $\beta_1, \beta_2, \beta_3, \beta_4$	723	971	692
At least one of $\beta_1, \beta_2, \beta_3, \beta_4 = 0$	0	0	0
At least one of $\beta_5, \beta_6 \neq 0$	277	29	308

(b) $n = 400$

Table 3: Section B.2: Result for Poisson regression with $n = 50$ and $n = 400$ and 1000 simulations. The table entries show the number of times (out of 1000) that a particular model was selected having the smallest IC value.

Based on the simulations, it is observed that BIC tends to pick the correct model more than AIC or TIC for both small (50) and moderate (400) sample sizes. This demonstrates the strong consistency of BIC (it almost surely selects the correct underlying model) All three methods were correctly able to conclude the non-zero effect of β_j, j = 1,...,4 at moderately larger sample size. The tendency of AIC to overfit is well known and is observed in table 3 at both sample sizes where it selects the model with $\beta_5 \neq 0$ or $\beta_6 \neq 0$ which is known to be redundant due to the simulation setup. Larger sample size leads to a small decrease in this number, however.

3 Part C

This part involves using a high dimensional data set (71 x 4089) to identify the subset of genes (out of 4088, 1 column is the response) that play a role in the production of vitamin B2. Since $p >> n$, a majority of the regression approaches will not be possible and approaches that can deal with the high dimensional nature of the problem will be necessary. For this part, we first split up the data into training ($n = 61$) and test sets ($n = 10$). The models are fit to the training set and the average of the squared prediction error (MSPE) on the test set is computed (i.e. the square of the difference(s) between the values predicted by the model and the observed values). A "good" model in this case should induce sparsity (the most important predictors are identified; important can be defined as coefficients having the largest point estimates) and having the lowest MSPE on the test set. Regression with the following penalties were tried: ridge ($\alpha = 0$), lasso ($\alpha = 1$), elastic net ($\alpha = 0.1, 0.2, \ldots, 0.9$) and adaptive lasso. Another thing to keep in mind is that multicollinearity is to be expected in our dataset with a very large number of variables and due to the nature of the problem.

For the adaptive lasso, we cannot use $\hat{\beta}_{\text{ols}}$ as starting values for the initial weights since these parameters cannot be estimated. However, Zou (2006, p.6) [1] mentions that any other estimator for $\hat{\beta}$ can be chosen. Thus, as initial weights we choose $\hat{\beta}_{\text{ridge}}$ and the corresponding weights then become $w_i = 1/|\beta_i|^\gamma$ where $\gamma = 0.5$ is chosen arbitrarily (re-

striction: $\gamma > 0$). This weight function ensures that coefficients with larger weights are penalized more and coefficients with smaller weights are penalized less. This is different from the lasso which penalizes all the coefficients equally.

The drawbacks of ridge and lasso are well known, namely that ridge regression does not induce sparsity as it continuously shrinks the coefficients towards zero without setting any of them to zero. Furthermore, if the variables are highly correlated, then lasso selects one out of the group of correlated variables and sets the rest to zero. In such a situation, elastic net usually gives more stable results.

For the models we fit, we are interested in two main parameters (apart from the usual regression parameters), namely the elastic net parameter $\alpha \in [0, 1]$ and the regularization parameter λ that keeps the size of the coefficients bounded. As mentioned above, α is evaluated for different values and for each *alpha*, we first pick a value for *lambda* using k-fold cross-validation ($k = 5$ in our case) on the training set that results in the smallest MSPE from a sequence of *lambda* values. Next, we extract the coefficients for the model at this *lambda* value and use these coefficients to calculate the MSPE on the test set. It should be noted that the k-fold CV is random as the folds are created randomly, however, if the model is truly able to capture the underlying relationship there should not be large variability across multiple runs. Table 4 presents the results from this process.

	alpha	MSPE	lambda_min	lambda_1se	num_coef (incl/ intercept)
1 (ridge)	0.00	0.19	7.15	31.66	4089
2	0.10	0.12	0.29	1.06	144
3	0.20	0.11	0.20	0.33	94
4	0.30	0.11	0.10	0.22	77
5	0.40	0.11	0.07	0.14	69
6	0.50	0.12	0.08	0.18	55
7	0.60	0.14	0.01	0.24	78
8	0.70	0.12	0.03	0.20	54
9	0.80	0.12	0.07	0.23	40
10	0.90	0.14	0.01	0.13	58
11 (lasso)	1.00	0.12	0.03	0.06	39
adaptive lasso	1.00	0.11	0.13	-	26

Table 4: Section C: MSPE on the test set for different values of the elastic net penalty $\alpha \in [0, 1]$. lambda_min is the value of the regularization parameter λ chosen by k-fold cross-validation. The last column displays the number of non-zero coefficients in the model selected for the corresponding alpha and lambda values.

Based on these results, we first observe that there is not that much difference in the MSPE for all the models except for the ridge regression fit. Since the prediction errors are very similar, we can perhaps pick parsimony as a criterion for selection and pick the model with the adaptive lasso penalty as our final model. There is quite a large difference between the number of coefficients in the adaptive lasso solution and the elastic net solutions. Whether this model is truly better (which would imply that we are choosing to ignore the small but non-zero effects of the other genes picked by the larger models) would require further assessment. However, on the basis of MSPE and Occam's razor, we pick the model using adaptive lasso as the final model. The coefficients from the adaptive lasso fit are provided in the Appendix 5. The interpretation of the coefficients might be straightforward as in the usual regression case, however, the variable names are not particularly informative for somebody lacking domain-knowledge.

References

[1] Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476), 1418-1429.

4 R Code

Note: The % > % operator (used in the following R snippets) from the package "magrittr" makes it easier to read code.

4.1 Part A

```
1  library(dplyr); library(magrittr); library(splines); library(SemiPar);
       library(mgcv)
2
3  fulldata.A = read.table("/home/ad/Desktop/KUL Course Material/Statistical
       Modelling/Assignment/Examdata2016-A.txt", header = T)
4  set.seed(475672)
5  rownumbers = sample(1:589, size = 150)
6  mydata.A = fulldata.A[rownumbers,]
7
8  mydata.A %<>% mutate(province = floor(mydata.A[,1]/10000) + 9*(floor(mydata
       .A[,1]/1000) == 21) + 8*(floor(mydata.A[,1]/1000) == 25))
9
10 fulldata.A %<>% mutate(province = floor(fulldata.A[,1]/10000) + 9*(floor(
       fulldata.A[,1]/1000) == 21) + 8*(floor(fulldata.A[,1]/1000) == 25))
11
12 mydata.A = mydata.A[sort.list(mydata.A$IntlMig_In),]
13
14 # Part A.1
15 province_name = c("Antwerp", "Flemish Brabant", "West Flanders", "East
       Flanders", "Hainaut", "Liege", "Limburg", "Luxembourg", "Namur", "
       Walloon Brabant", "Brussels")
16
```

```r
mydata.A %>% group_by(province) %>%  summarise(birthsPer1000 = mean(y),
    AvgIntl_in = mean(IntlMig_In))

mydata.A %>% group_by(province) %>% count(province)

# filter province 2 and 11
temp_df = filter(mydata.A, province %in% c(2,11)) %>%
  dplyr::select(y, IntlMig_In, province) %>%
  mutate(province = as.factor(province)) %>% arrange(province)

# wilcoxon test for testing the means of crude birth rate
wilcox.test(formula = y ~ province, conf.int = TRUE, data = temp_df)
wilcox.test(formula = IntlMig_In ~ province, conf.int = TRUE, data = temp_
    df)

# comparing against the whole dataset
fulldata.A %>% group_by(province) %>% count(province)

# filter province 2 and 11
temp_df_full = filter(fulldata.A, province %in% c(2,11)) %>%
  dplyr::select(y, IntlMig_In, province) %>%
  mutate(province = as.factor(province)) %>% arrange(province)

# wilcoxon test for testing the means of crude birth rate
wilcox.test(formula = y ~ province, conf.int = TRUE, data = temp_df_full)
wilcox.test(formula = IntlMig_In ~ province, conf.int = TRUE, data = temp_
    df_full)

# Part A.2
fit1 = spm(y ~ f(IntlMig_In), family = "poisson")
summary(fit1); plot(fit1); points(IntlMig_In, log(y_temp))

# can put knots at 50, 400, 1300 and 9500
fit_part = glm(y ~ IntlMig_In + pmax((IntlMig_In - 50),0) + pmax((IntlMig_
    In - 400),0) + pmax((IntlMig_In - 1300),0) + pmax((IntlMig_In - 9500)
    ,0), family = Gamma(link = "log")); summary(fit_part); plot(fit_part)

drop1(fit_part, test = "LRT")
lrtest(update(fit_part, . ~ . - pmax((IntlMig_In - 50),0) - pmax((IntlMig_
    In - 9500),0)), fit_part)

# plotting the picewise linear model on the spline fit
plot(fit1, ylim = c(2,3.2), xlab = "Number of Incoming International
    Migrants", ylab = "log(Expected birth rate per 1000)", lwd = 2)
points(IntlMig_In, log(y))
lines(IntlMig_In, log(fit_part$fitted.values), col = "green", lwd = 3)
abline(v = c(50, 400, 1300, 9500), lty = 2, lwd = 2)
legend(2800, 2.4, legend = c("Penalized Spline (33 knots)", "Linear spline
    w/ 4 knots (no penalization)"), col = c("black", "green"), lwd = c(2,3)
    , lty = c(1,1)); title(main = "knots at 50, 400, 1300, 9500")

BIC(fit_part)
BIC(update(fit_part, . ~ . - pmax((IntlMig_In - 50),0) - pmax((IntlMig_In -
     9500),0)))
```

```
61
62 # Part A.3
63 fit_additive = spm(y ~ f(Income, basis = "trunc.poly") + f(Zero_Income,
       basis = "trunc.poly"), family = "poisson"); summary(fit_additive); par(
       mfrow = c(1,2)); plot(fit_additive)
64
65 # Part A.4
66 fit_glm_add = glm(y ~ Income + Zero_Income, family = Gamma("log"))
67 fit_glm_full = glm(y ~ Income + Zero_Income + Income:Zero_Income + I(Income
       ^2) + I(Zero_Income^2), family = Gamma("log"))
68 summary(fit_glm_add); summary(fit_glm_full)
69
70 library(lmtest)
71
72 lrtest(fit_glm_add, fit_glm_full)
73 drop1(fit_glm_full, scope = ~Income:Zero_Income + I(Income^2) + I(Zero_
       Income^2), test = "LRT")
74
75 # Part A.5
76 # fitting splines using spm
77 fit_all = spm(y ~ Income + Zero_Income + f(Density) + f(Marriages) + f(
       Divorces) + f(Deaths) + f(IntMig_In) + f(IntMig_Out) + f(IntlMig_In) +
       f(IntlMig_Out) + f(Nationality_In) + f(Population_Dec), family = "
       poisson", random = ~1, group = province); summary(fit_all); plot(fit_
       all, jitter.rug = TRUE)
78
79 # fitting splines using gam
80 fit_all_gam = gam(y ~ s(Income) + s(Zero_Income) + s(Density) + s(Marriages
       ) + s(Divorces) + s(Deaths) + s(IntMig_In) + s(IntMig_Out) + s(IntlMig_
       In) + s(IntlMig_Out) + s(Nationality_In) + s(Population_Dec) + s(
       province), family = Gamma("log")); summary(fit_all_gam); plot(fit_all_
       gam)
81
82 # fitting spm with one nonparamteric
83 fit_all_spm = spm(y ~ Income + Zero_Income + Density + Marriages + Divorces
       + Deaths + IntMig_In + IntMig_Out + f(IntlMig_In) + IntlMig_Out +
       Nationality_In + Population_Dec, family = "poisson", random = ~1, group
       = province)
84 summary(fit_all_spm)
85
86 # fitting parametric with quadratic and cubic
87 fit_final = glm(y ~ Income + Zero_Income + Density + Marriages + Divorces +
       Deaths + IntMig_In + IntMig_Out + IntlMig_In + I(IntlMig_In ^ 2) + I(
       IntlMig_In ^ 3) + IntlMig_Out + Nationality_In + Population_Dec + as.
       factor(province), family = Gamma("log")); summary(fit_final)
88
89 drop1(fit_final, test = "LRT")
90
91 # reduced model with a subset of predictors
92 fit_reduced = glm(y ~ Income + Deaths + IntMig_In + IntMig_Out + IntlMig_In
       + I(IntlMig_In ^ 2) + I(IntlMig_In ^ 3) + IntlMig_Out + Population_Dec
       , family = Gamma("log")); summary(fit_reduced)
93
94 lrtest(fit_reduced, fit_final)
```

```
95
96 # model with knots instead of quadratic and cubic effects
97 fit_reduced2 = glm(y ~ Income + Deaths + IntMig_In + IntMig_Out + IntlMig_
       In + pmax((IntlMig_In - 400),0) + pmax((IntlMig_In - 1300),0) + IntlMig
       _Out + Population_Dec, family = Gamma("log")); summary(fit_reduced2)
98
99 BIC(fit_reduced); BIC(fit_reduced2)
```

4.2 Part B

```
1  set.seed(475672)
2  # num of samples
3  n = 400 # 50, 400
4  # number of simulations
5  nsim = 1000
6  # list to store the final results
7  final50 = list()
8  final400 = list()
9
10 # simulation code
11 for (i in 1:nsim) {
12
13 x2 = runif(n, -1, 1); x3 = runif(n, -1, 1); x4 = runif(n, -1, 1); x5 =
       runif(n, -1, 1); x6 = runif(n, -1, 1)
14 mu_i = (1/3) + ((2/3) * x2) + x3 + ((4/3) * x4) + (0 * x5) + (0 * x6)
15 y = rpois(n, exp(mu_i))
16 des_mat = data.frame(Int = 1, x2, x3, x4, x5, x6)
17 # creating the combinations of variables
18 L = 5
19 combinations = function(L){
20   comb = NULL
21   for (i in 1:L) {comb = rbind(cbind(1,comb),cbind(0,comb))}
22   return(comb)
23 }
24
25 # models to be fit
26 combination = cbind(1, combinations(5))
27 aic = c()
28 bic = c()
29 tic = c()
30 form = c()
31
32 for (k in seq_along(1:2 ^ L)) {
33
34   temp_df = des_mat[,combination[k,] > 0]
35
36   # this block is included because when k = 2^L only the intercept term is
       in the model in which case temp_df returns a vector and not a data
       frame so we can fit this manually for the last iteration; stupid edge
        cases
37   if (k == 2 ^ L) { formulae = "y ~ 1"
38     fit1 = glm(formulae, family = poisson)
```

```
  } else{
        formulae = temp_df %>% names() %>% paste(collapse = " + ") %>%
      paste("y ~ 0 + ", ., sep = "")
    fit1 = glm(formulae, data = temp_df, family = poisson)
  }

  aic[k] = AIC(fit1)
  bic[k] = BIC(fit1)
  form[k] = formulae

  # calculating TIC
  # use y = y and x = temp_df
  temp_df %<>% as.matrix()
  beta_est = fit1$coefficients %>% as.matrix()
  mu_i_est = (temp_df %*% beta_est) %>% exp()

  first_deriv = (temp_df * as.vector((y - mu_i_est))) %>% as.matrix()

  Kmatrix = (t(first_deriv) %*% first_deriv)/n # p x p matrix

  # I(\beta) = (X^T)(D(\beta))(X)
  # where D(\beta) = the diagonal matrix with mu_i_est on the diagonal
      elements
  # (Observed) Fisher Information Matrix - p x p matrix
  Jmatrix = (t(temp_df) %*% diag(as.vector(mu_i_est)) %*% temp_df)/n

  penaltyTIC = (solve(Jmatrix) %*% Kmatrix) %>% diag() %>% sum()

  tic[k] = (-2 * c(logLik(fit1))) + (2 * penaltyTIC)
}

results = data.frame(Model = form, AIC = aic, BIC = bic, TIC = tic)
bestAIC = results %>% filter(AIC == min(AIC)) %>% select(Model, AIC)
bestBIC = results %>% filter(BIC == min(BIC)) %>% select(Model, BIC)
bestTIC = results %>% filter(TIC == min(TIC)) %>% select(Model, TIC)

if (n == 50) {
  final50$modelAIC[i] = as.character(bestAIC$Model)
  final50$AIC[i] = bestAIC$AIC
  final50$modelBIC[i] = as.character(bestBIC$Model)
  final50$BIC[i] = bestBIC$BIC
  final50$modelTIC[i] = as.character(bestTIC$Model)
  final50$TIC[i] = bestTIC$TIC
}

if (n == 400) {
  final400$modelAIC[i] = as.character(bestAIC$Model)
  final400$AIC[i] = bestAIC$AIC
  final400$modelBIC[i] = as.character(bestBIC$Model)
  final400$BIC[i] = bestBIC$BIC
  final400$modelTIC[i] = as.character(bestTIC$Model)
  final400$TIC[i] = bestTIC$TIC
}

```

```
92  stopifnot(n != 50 || n != 400)
93  }
94
95  ## analyzing the results
96  # n = 50
97  table(final50$modelAIC)
98  table(final50$modelBIC)
99  table(final50$modelTIC)
100
101 # n = 400
102 table(final400$modelAIC)
103 table(final400$modelBIC)
104 table(final400$modelTIC)
```

4.3 Part C

```
1   library(glmnet)
2   library(xtable)
3
4   fulldata.C = read.table("/home/ad/Desktop/KUL Course Material/Statistical
        Modelling/Assignment/Examdata2016-C.txt", header = T, sep = ",")
5   set.seed(475672)
6   rownumbers = sample(1:71,61,replace = F)
7   mytraining = fulldata.C[rownumbers,]
8   mytest = fulldata.C[-rownumbers,]
9
10  # ridge regression for the coefficients to be used in adaptive lasso
11  ridge_cv = cv.glmnet(x = data.matrix(mytraining[,-1]), y = data.matrix(
        mytraining[,1]), nfolds = 5, alpha = 0, type.measure = "mse")
12  plot(ridge_cv)
13  ridge_cv$lambda.min
14
15  # variables for elastic net
16  alpha = seq(0, 1, 0.1)
17  mspe = c()
18  lambda_min = c()
19  lambda_1se = c()
20  num_coef = c()
21
22  for (i in seq_along(alpha)) {
23
24    # fitting the model to the training set
25    mod_cv = cv.glmnet(x = data.matrix(mytraining[,-1]), y = data.matrix(
          mytraining[,1]), nfolds = 5, alpha = alpha[i], type.measure = "mse")
26
27    plot(mod_cv, main = paste("alpha = ", alpha[i], sep = ""))
28
29    lambda_min[i] = mod_cv$lambda.min
30    lambda_1se[i] = mod_cv$lambda.1se
31
32    # getting the predictions on the test set
```

```
33    pred = predict(mod_cv, newx = data.matrix(mytest[,-1]), s = "lambda.min")
          %>% as.vector()
34
35    # mean of squared prediction errors
36    mspe[i] = (mytest[,1] - pred) ^ 2 %>% mean()
37
38    # number of coefficients in the selected model
39    select_coef = coef(mod_cv, s = "lambda.min") %>% as.matrix()
40    num_coef[i] = select_coef[select_coef != 0,] %>% length()
41
42  }
43
44  beepr::beep(3)
45
46  results = data.frame(alpha, MSPE = mspe,  lambda_min, lambda_1se, num_coef)
47  results
48
49  # adaptive lasso using glmnet and penalty factor
50  ridge_coef = coef(ridge_cv, s = "lambda.min") %>% as.matrix()
51  adapt_wt = 1/(abs(ridge_coef)) ^ 0.5
52
53  ada_lass = cv.glmnet(x = data.matrix(mytraining[,-1]), y = data.matrix(
        mytraining[,1]), alpha = 1, penalty.factor = adapt_wt, nfolds = 5, type
        .measure = "mse")
54  plot(ada_lass)
55  print(ada_lass)
56
57  # 0.1344 is the smallest lambda value chosen through print() and explains
        92% of the variability in the response
58  pred = predict(ada_lass, newx = data.matrix(mytest[,-1]), s = 0.1344) %>%
        as.vector()
59  (mytest[,1] - pred) ^ 2 %>% mean()
60
61  ada_coef = coef(ada_lass, s = 0.1344) %>% as.matrix()
62
63  # which covariates are selected in the final model
64  ada_coef[ada_coef != 0,] %>% round(digits = 4)
```

5 Appendix

Coefficients from the final model estimated in Part C.

Variables	Coefficients
(Intercept)	-13.5339
x.AADK_at	0.6148
x.CARA_at	-0.1134
x.DEGA_at	0.1720
x.FLHP_at	0.1838
x.LYSC_at	-0.2296
x.RIBT_at	0.2358
x.XHLA_at	0.0074
x.XKDC_at	0.5901
x.XSA_at	-0.4354
x.YCLC_at	0.0861
x.YDBI_at	0.0517
x.YDBM_at	-0.1313
x.YDDK_at	-0.0200
x.YDDM_at	-0.0489
x.YFHE_r_at	0.4319
x.YFIQ_at	0.1879
x.YHDS_r_at	0.1243
x.YHDT_at	0.0601
x.YISU_at	0.1121
x.YOAC_at	-0.3000
x.YRPE_at	-0.1700
x.YURR_at	0.2297
x.YWPC_at	-0.0944
x.YXLC_at	-0.3236
x.YYCR_at	-0.3386

Table 5: Section C: Estimated coefficients for the final model selected using the adaptive lasso.