Analysis of the Correlation between Diet and COVID-19 Cases by Country

Zeng Fung Liew (lzfliew@ucdavis.edu)

December 12, 2020

Abstract

The goal of this analysis is to find out how a country's diet correlates with its number COVID-19 cases. With different food cultures across the world, it would be interesting to see what are the food categories that can best predict a country's rate of confirmed cases, deaths, recovery cases, and active cases respectively. It turns out that there are some food categories which, if the intake is high (or too low), highly correlates with the high percentage of COVID-19 cases per capita.

The next thing that was investigated was the fitting of a suitable linear model for the purpose of COVID-19 case predictions. Due to the large amount of predictor variables in this data, there were a few preliminary steps taken to eliminate some of them prior to the fitting of models, some of which include eliminating predictor variables of the same food type that have high correlations with each other, removing predictor variables with (almost) no distribution, and deleting predictor variables with large variance inflation factors (VIF), ie. predictor variables that are highly inter-correlated with the rest of the predictor variables. In the model selection process, the models were started off without any predictor variables and the forward stepwise regression method was applied with respect to both the AIC and BIC criteria. The resulting models were then analyzed based on their C_p, $Press_p$, adjusted R^2, and mean square prediction error (MSPE) criteria to determine the model that generalizes better. Then, possible outliers were analyzed using the Cook's distance plot and subsequently removed before the selection of the best prediction model.

1 Introduction

The analysis of this data set was motivated by the rising COVID-19 cases across the world. Due to the different food and diet cultures across countries, it is interesting to find if any correlation exist between a country's typical diet and its number of COVID-19 confirmed, death, recovered, and active cases. Obviously, the correlations shown in this data analysis (if any) are not indications causation effect, ie. having certain food in large amount does not lead to a higher chance of contacting COVID-19, but it could be an indicator of strength of the general population's immune system, which could have an effect on the spread of COVID-19 in the country.

The data set used in this analysis was found on Kaggle, created and maintained by Marie Ren. It comes in four data frames containing 32 columns each, where each data frame takes a different measurement of the food categories, ie. fat quantity, food quantity, food energy, and protein quantity. The quantitative variables are represented in terms of percentages, eg. the percentage of fat content in eggs in a diet by country, or the percentage of the population of confirmed cases of COVID-19. However, not all of the columns will be used in this data analysis. As the goal of this analysis is to find the correlations between diet and COVID-19 cases, variables like obesity and undernourished rate, and country populations were ignored. Only the columns that describe predictor variables (food categories), response variables (confirmed, death, recovered, active cases), and countries were retained. This data analysis was started after combining the four data frames into one large data frame containing 170 rows and 100 columns.

The aim of this data analysis is to figure out the food categories that can best predict the number of confirmed, death, recovered, and active cases of COVID-19 in the world. It would also be interesting to see if

these food categories are the same across each of those response variables, ie. if the food categories that can predict the rate of recovered can also predict the rate of death cases.

This data analysis was done on the most recently updated data set at the time of writing, which was on December 2, 2020. Changes in the COVID-19 situations across countries could result in different analysis results.

2 Methods and Results

2.1 Exploratory Data Analysis

As the main goal of this data analysis is to fit the best possible Ordinary Least Squares linear models to predict the rate of confirmed, death, recovered, and active cases, it is important to make sure that its assumptions are satisfied prior to the model fitting step. Some of the important assumptions that have to be satisfied are:

- Normal distribution of the response variables and the constant variance among residuals. If this isn't the case in the given data, a transformation of variables has to be carried out.
- Any two responses are uncorrelated. This is assumed to be true on the given data set.
- Linearity between each predictor variable with each response variable.

Additionally, due to the large amount of predictor variables in this data set (92 food categories and measurements!), multicollinearity is bound to exist, which could severely hurt the model fitting process. To counter this issue, a preliminary variable elimination step was taken to remove "redundant" food categories/measurements.

Prior to the fitting of any model, it is also important to check if the data contains any missing values. In this data set, there were no missing values among the predictor variables, but there were some missing values in the response variables, ie. there were no reports on the confirmed, death, recovered, and active case rates on some of the countries in this data set. After the removal of those rows, only 163 rows of data remained in the Confirmed, Death, and Recovered cases data frames, while only 161 rows of data remained in the Active cases data frame.

2.1.1 Preliminary Variable Elimination

There were three criteria used to determine which predictor variables should be removed prior to model fitting. The goal here is to avoid the issue of multicollinearity in the later stages.

The first thing that was done was to select "representative" variables among the same food categories with different measurements. This is not only because of the multicollinearity issue caused by large correlation values, but also because if one of the variables is a good predictor of COVID-19 cases, the other highly correlated variables do not help much in further explaining the model. An example of the selection of "representative" variables is shown in Figure 1. In this step alone, the total number of predictor variables have been reduced from 92 to 51.

Another thing that was done to reduce the number of predictor variables prior to the model fitting step is to remove predictor variables with (close to) no distribution. Predictor variables like the fat content in alcoholic beverages can be removed as shown in Figure 2 since it wouldn't explain anything about the variability of COVID-19 cases across the world. In this step, another 6 predictor variables have been removed from the pool of predictor variables.

Lastly, predictor variables with high variance inflation factors (VIF) are also subsequently removed. While these variables might not be pairwise correlated with each predictor variable, they are highly intercorrelated with the rest of the predictor variables, which can also lead to multicollinearity among predictor variables. Some of the predictor variables with the largest VIF values that were removed are shown in Table 1. With those predictor variables removed, the number of predictor variables left prior to the start of the model fitting step is 41.

2.1.2 Variable Transformation

To determine if the response variables require transformation, their histograms are plotted as shown in Figure 3. If those histograms do not resemble normal distributions, it means that the first assumption stated in the previous section are not satisfied. To counter this issue, the Box-Cox procedure (Figure 4) is carried out to determine the best transformation of the response variables. In this case, the best transformation for all the response variables are all log transformations. This is subsequently carried out prior to the start of the model fitting process and the distribution of the transformed response variables are shown in Figure 5.

To determine if the predictor variables require transformation, each of the predictor variables were plotted against each response variables. A transformation of a predictor variable is needed if there is a significant non-linear relationship between the predictor variable and a transformed response variable. Since this is not the case in this data set, no transformation of the predictor variables is necessary.

2.1.3 Top Correlated Food Categories with the Number of Cases

Table 2 shows each response variable's top 5 correlated predictor variables. The goal of learning about the correlations between the response and predictor variables here is to observe how closely related they are. Note that the top pairwise correlated variables have absolute correlation values between 0.4 to 0.5, indicating that the number of COVID-19 cases has some correlation with some food categories, ie. the protein quantities of miscellaneous food and tree nuts, the calorie content in eggs, and animal fat.

2.2 Model Fitting and Validation

After gaining a slight understanding of the summary of the distribution of the data and removing redundant variables, the next step was to fit models that are able to do great jobs in predicting the rate of confirmed, death, recovered, and active cases respectively. There are two types of procedures to search for good models: the best subset selection and the stepwise regression methods. The best subset selection is guaranteed to work better since it parses through all the possible models with the given set of predictor variables, as compared to the stepwise regression method which is a greedy search. However, the best subset selection method is not a viable option in this case due to the large number of predictor variables. If the aim is to find the best model containing only linear terms and possible interaction terms, there would be a total $41 + \binom{41}{2} = 861$ variables to choose from, and a total of 2^{861} models to choose from. Not only is the process going to take infinitely long, the process is also not feasible since the number of variables could end up being more than the number of rows in our data.

Instead, the approach taken in this data analysis was to find the best model using the stepwise regression starting from the null model, ie. $\hat{Y}_i = \hat{\beta}_0$. Two different models were then built using the AIC and BIC criterion respectively. The AIC criterion, known to be less conservative as compared to the BIC criterion, produced larger models when fitting models for all the response variables, ie. confirmed, death, recovered, and active cases.

Before the model fitting procedure was carried out, a (70%-30%) random splitting of data was carried out into training and validation sets for all four data frames. Only the training sets will be used for the model fitting process, whereas the validation sets will be used to validate the prediction abilities of the "best" model acquired from the training sets.

2.2.1 Internal Validation

To determine the best model for each of the response variables, four tables were created as shown in Table 3 comparing the predictive abilities and the model bias and variability of the AIC and BIC selected models. The problem faced in calculating one of the metrics, Mallow's C_p, was the inability to fit a full model, ie. a model with all predictor variables and their two-way interactions, due to the large number of predictor variables here. This issue was handled by assumming that the model that contains all the variables of both the AIC and BIC models as the full model. In the case of model fittings for confirmed and recovered cases,

the AIC models were considered as the full models since they contain all the predictor variables present in their BIC counterparts.

For the confirmed cases models (Table 3(a)), the SSE and MSE values are found to be quite small in both the AIC and BIC models, which suggests a very good model fit in this case. This is backed up by their respectively R_a^2 values. With the R_a^2 being higher than 0.6, it suggests that a huge variability of the number confirmed cases can be explained by the selected predictor variables. The $Press_p$ values for models are also very close to their respective SSE values, which suggests strong predictive abilities of both models. What allows for the AIC model to edge out in this case is its C_p value being significantly smaller than its BIC counterpart, suggesting very little model bias.

A similar conclusion can be made based on the AIC and BIC models for active cases, shown in Table 3(d). However, the difference between the models for active cases and the models for confirmed cases here is that they have higher R_a^2 values while also having significantly higher SSE, MSE, and $Press_p$ values. This suggests that while the predictor variables can explain the number of active cases better, its predictive ability is however significantly lower than that of the confirmed cases models.

On the other hand, the models explaining the rate of recovered cases (Table 3(c)) did not perform well. They all have extremely large SSE and MSE values, and very small R_a^2 values, suggesting that diet does not really explain the number of recovered cases well.

The AIC and BIC models for the number of death cases (Table 3(b)) is somewhat interesting. Unlike the aforementioned models, the difference in sizes between these two models is huge (a difference of 30 predictor variables/interaction terms). Here, the SSE, MSE, R_a^2, and C_p values are very significantly different. This can be caused by the overfitting of the AIC model and the underfitting of the BIC model. However, the $Press_p$ values are not much different between the two models. As the $Press_p$ value for the AIC model is very far away from the SSE, while the opposite is true for the BIC model, it seems that the BIC model has a better predictive ability.

Based on the results obtained so far, it seems that all the AIC models are the better choices. However, a secondary step of model validation on the validation data is required to ensure that the models are not overfitting onto the training data.

2.2.2 External Validation

The goal of this procedue is to ensure that the models do not only work well on the training data, ie. the models selected should be able to generalize the trend of cases across all the countries in the world instead of the ones in the training data. The metrics used in the external validation process is simple, which is comparing the mean squared prediction error (MSPE) of the validation data with that of the training data. Mathematically, MSPE is defined as

$$MSPE = \frac{\sum_{j=1}^{m}(Y_j - \hat{Y}_j)^2}{m}$$

where m is the number of data either in the validation set or the training set. Essentially the MSPE of the training data can be thought of as SSE/n. The results of the MSPE comparisons between the AIC and BIC models are shown in Table 4.

From the table, it is obvious that in the confirmed cases models and active cases models, the $MSPE_{valid}$ produced by the BIC models are far closer to their respective $MSPE_{train}$ than their AIC counter parts. This is due to the result of overfitting in the training data set. In other words, using the AIC models on the training data set produces more accurate predictions only on the training data, not in general. In selecting the best model, however, requires the model to generalize well, hence the BIC models are selected in both of these cases.

In fact, this scenario also happens at an even bigger margin in the death cases models. Here, the $MSPE_{valid}$ is significantly larger than the $MSPE_{train}$ for the AIC model, whereas the opposite is true for the BIC model. This is due to the severe overfitting of the AIC model and the severe underfitting of the BIC model, which are both not ideal. However, the BIC model is selected as it is able to generalize better than

the AIC model.

As for the recovered cases models, since the $MSPE_{train}$ values are similar for both the AIC and BIC models, the model with the smaller $MSPE_{valid}$ value is selected since it is able to generalize better. Therefore, the BIC model is once again selected.

In summary, all the BIC models seem to generalize better than their AIC counterparts. What remains right now is that these BIC models satisfy the necessary model assumptions before fully concluding the best model.

2.3 Model Diagnostics

The goal of this procedure is to ensure the following:

- The BIC model residuals have equal variance.
- The BIC model residuals are normally distributed.
- Analyze possible outliers and decide if it is necessary to have them removed.

The first two assumptions above can be analyzed using the residuals vs fitted values plot and the QQ-plots respectively, as shown in Figure 6. From the set of plots, we find that the confirmed cases and active cases models are satisfy those assumptions reasonably well. Bar a few possible outliers, we can simply assume that the residuals of those models have equal variances and are normally distributed.

On the other hand, the same cannot be said about the recovered cases and death cases plots. Interestingly, the residuals vs fitted values plot for death cases seems to come in a pattern of two straight lines. Its QQ plot also showed a significant deviation from the normal line at theoretical quantiles less than -1. A similar situation also arises from the recovered cases model. This could be the reason why the model fit for the confirmed cases and active cases models are significantly better than the death cases and recovered cases models.

Moving on, to find the possible outliers of each model, the residuals, leverage values and Cook's distance are analyzed. One way to do so is by looking at the leverage vs residuals plots, shown in Figure 7. Essentially, points with large leverage values indicate outlying predictor variables, and points with large absolute values of standardized residuals indicated outlying response variables. If any of these points lie outside of the red dotted lines (aka. the line of Cook's distance), it becomes a strong indication that the points are outliers. Based on the figure, it can be seen that case 165 (Vanuatu) is a significant outlier for both the recovered cases and active cases models, whereas case 15 (Sri Lanka) is also a significant outlier for the recovered cases models.

Another thing that can be checked is the Cook's distance plot, as shown in Figures 8 to 11. Those plots make it easier to determine if a data point has a significantly huge Cook's distance value. Typically, a Cook's distance $D_i > \frac{4}{n-p}$ is used to indicate whether a data point i is a potential influential case. In more conservative cases, $D_i > 1$ is used as the cutoff instead. However, due to the small amount of data here, setting the cutoff to be $\frac{4}{n-p}$ means too many data points will be thrown away, whereas a cutoff of 1 would be too conservative. Instead, the "cutoff" used in this analysis is based off the Cook's distance plots, where data points that look to have Cook's distance significantly larger than the rest of the data points will be considered as potential outliers.

With these potential outliers in mind, the next thing to do is test if they really do heavily influence their respective models. This is done by calculating the percentage change between the fitted values based on the models with and without the potential outliers, ie.

$$\alpha_i = \frac{Y_i - Y_{i,(-J)}}{Y_i} \times 100\%, \text{ where } J \text{ are potential outliers}$$

Using the above equation, it can be found that the percentage changes between fitted values based on models with and without potential outliers are very significant (up to 30% for confirmed cases model, up to 164.9% for death cases model, and up to 260.9% for recovery cases model). Therefore, the potential outliers

were subsequently removed. On the other hand, this was not the case for the active cases model (close to 0% change) and hence the potential outliers were retained.

After eliminating the outliers from the models, it is time to move into the last step of this data analysis, which is fitting the final models.

2.4 Final Models

The final step of this data analysis is simply combining the training data (without outliers) and the validation data, and refit the data into the same models. The final models are shown as follows:

Confirmed cases model:

$$Y = \exp(-2.8 + 0.2X_1 + 0.17X_2 - 0.22X_3 - 0.26X_4 - 11.71X_5 - 0.22X_6 + 0.46X_7 + 2.67X_8 - 1.46X_6X_8)$$

where Y is rate of confirmed cases of COVID-19, X_1 is Miscellaneous (Protein), X_2 is Sweeteners (Calories), X_3 is Meat (Calories), X_4 is Pulses (Calories), X_5 is Stimulants (Protein), X_6 is Oilcrops (Calories), X_7 is Fruits (Protein),and X_8 is Eggs (Calories). The R_a^2 value for this model is 0.5631, which suggests that this model is a relatively good predictor of COVID-19 case rates of each country.

Death cases model:

$$Y = \exp(-14.86 + 0.53X_1 + 0.24X_2 + 3.12X_3)$$

where Y is rate of death cases of COVID-19, X_1 is Miscellaneous (Protein), X_2 is Vegetable Oil (Fat), and X_3 is Treenuts (Calories). The low R_a^2 value of 0.18 in this model indicates that diet only explains about 20% of the variability of COVID-19 deaths.

Recovery cases model:

$$Y = \exp(0.68 - 2.14X_1 - 90.13X_2 + 81.44X_1X_2)$$

where Y is rate of recovered cases of COVID-19, X_1 is Oilcrops (Calories), and X_2 is Vegetal Products (Protein). The low R_a^2 value of 0.21 in this model once again indicates that diet is not a good predictor of COVID-19 recovery.

Active cases model:

$$Y = \exp(-4.33 - 1.67X_1 + 0.21X_2 + 0.19X_3 + 0.21X1X2)$$

where Y is rate of active cases of COVID-19, X_1 is Oilcrops (Calories), X_2 is Miscellaneous (Protein), and X_3 is Alcoholic Beverages (Quantity). The low R_a^2 value of 0.58 in this model suggests that diet can explains about 58% of the variability of COVID-19 active cases.

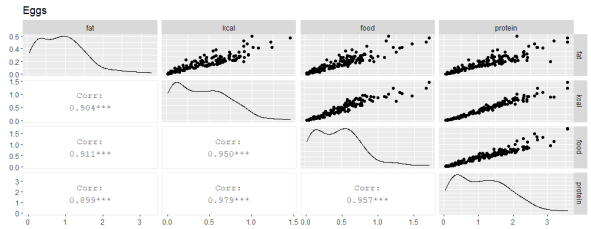
3 Conclusion

In summary, a country's COVID-19 confirmed and active cases can somehow be explained relatively well by food categories such as the calorie contents of oilcrops, and the protein content in infant food and miscellaneous food. On the other hand, the same cannot be said about the death and recovered cases. This could be due to the fact that these models do not satisfy the neccessary model assumptions of having equal variance and normally distributed residuals. However, it is also important to note that active cases are simply cases that have not had an outcome, and hence the model on active cases should only be taken as a grain of salt, or rather a model that follows closely with the model of confirmed cases. In short, the model predicting the rate of confirmed cases is the only reliable model obtained from this analysis.

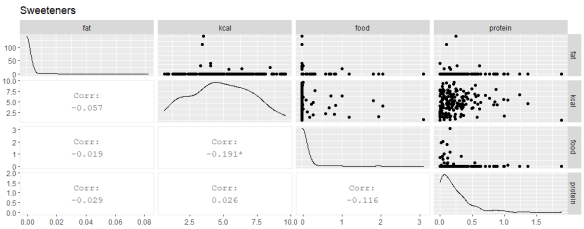
However, recall that this model only talks about the correlation between food categories and the rate of confirmed cases. There is no evidence to suggest that a country's diet has an effect on the spread of COVID-19. Additionally, there are also many other factors causing the spread of COVID-19 that are totally uncorrelated with diet, eg. how active the general public are, the preventive measures implemented by the countries, density of population etc.

4 Appendices

4.1 Appendix 1: Figures and tables



((a)) The fat, calorie, food, and protein quantities of eggs are highly correlated. Hence, only the calorie quantity in eggs is selected as the "representative" variable.



((b)) There does not seem to be any pairwise correlation between any of the measurements of sweeteners. Hence, all 4 measurements are retained.

Figure 1: Scatterplot Matrices of Food Categories with Different Measurements

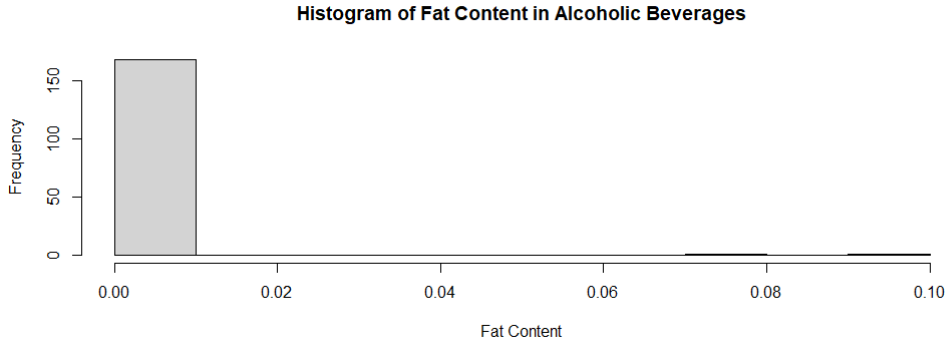
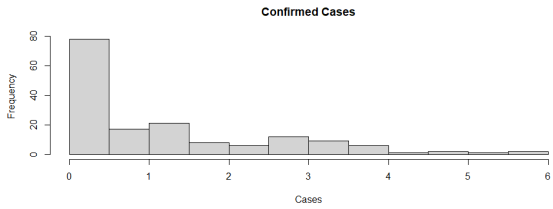


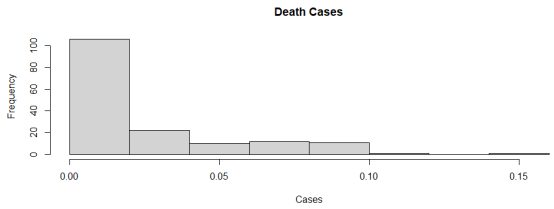
Figure 2: Variables like the fat quantity in alcoholic beverages do not have distributions, and thus can be removed.

Table 1: The variables above have extremely high VIF values, indicating very high intercorrelation with the rest of the predictor variables, and should be subsequently removed.

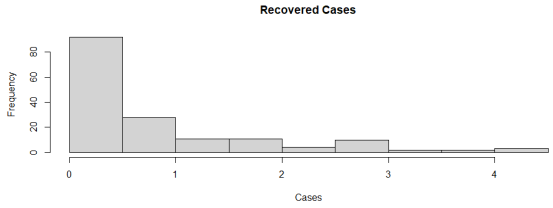
Variable	VIF
Animal Fats (Food Quantity)	2,096,672.147
Vegetables (Food Quantity)	2,094,971.233
Animal Products (Calories)	238,996.825
Vegetable Products (Calories)	237.921.552



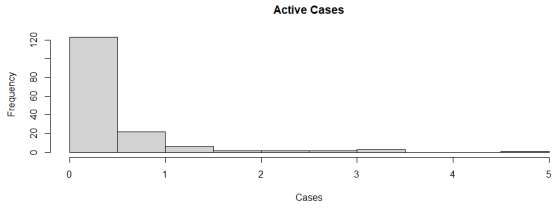
((a)) Histogram of Confirmed Cases



((b)) Histogram of Death Cases

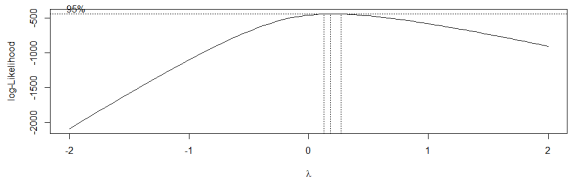


((c)) Histogram of Recovered Cases

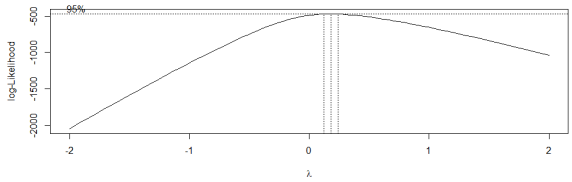


((d)) Histogram of Active Cases

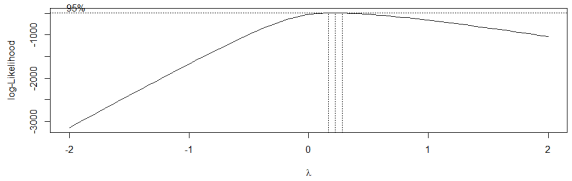
Figure 3: Histograms of Response Variables



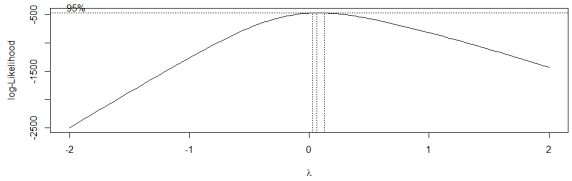
((a)) Box-Cox of Confirmed Cases



((b)) Box-Cox of Death Cases

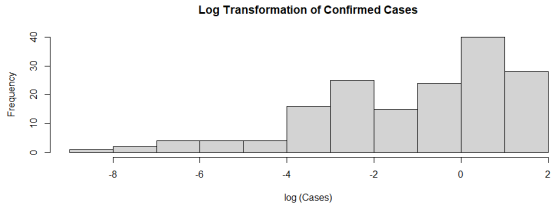


((c)) Box-Cox of Recovered Cases



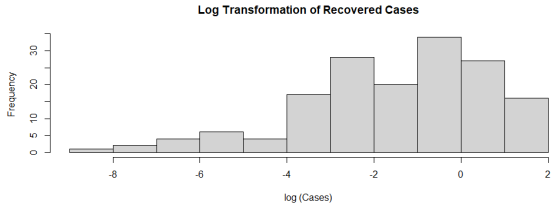
((d)) Box-Cox of Active Cases

Figure 4: Based on the above plots, it can be seen that the log transformation is the best way to go for all of the response variables.

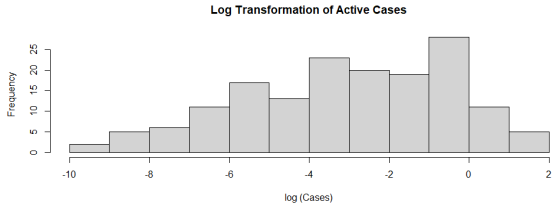


((a)) Histogram of log transformed Confirmed Cases

((b)) Histogram of log transformed Death Cases



((c)) Histogram of log transformed Recovered Cases



((d)) Histogram of log transformed Active Cases

Figure 5: After carrying out the log transformation on each of the response variable, their histograms are significantly closer to being normally distributed. Note: extremely small values like 10^{-11} were added into each data point of death, recovered and active cases due to the existence of zero values.

Table 2: Top 5 highly correlated predictor variables with their respective response variables. Note: The letter codes are P - protein, Q - food quantity, C - calories, F - fat.

Confirmed		Deaths		Recovered		Active	
Misc. (P)	0.59	Misc. (P)	0.45	Stimulants (F)	0.45	Misc. (P)	0.5
Treenuts (P)	-0.52	Eggs (C)	0.43	Misc. (P)	0.42	Animal Prod. (Q)	0.45
Eggs (C)	0.49	Treenuts (P)	-0.42	Treenuts (P)	-0.39	Animal Fat (F)	0.43
Animal Fat (F)	0.47	Animal Fat (F)	0.39	Eggs (C)	0.38	Treenuts (P)	-0.42
Stimulants (F)	0.44	Cereal (F)	-0.35	Offals (P)	-0.32	Veg. Prod. (P)	0.42

Table 3: Analysis of the AIC and BIC models to determine the best model

	AIC Model	BIC Model
p	16	10
SSE	150.12	189.24
MSE	1.53	1.82
R^2	0.73	0.66
R_a^2	0.69	0.63
C_p	16	29.54
$Press_p$	218.35	242.87

(a): Confirmed cases models

	AIC Model	BIC Model
p	34	4
SSE	856.33	3774.95
MSE	10.7	34.32
R^2	0.82	0.2
R_a^2	0.74	0.17
C_p	35.5	253.29
$Press_p$	3978.05	4017.27

(b): Death cases models

	AIC Model	BIC Model
p	5	4
SSE	2203.47	2281.23
MSE	20.22	20.74
R^2	0.3	0.27
R_a^2	0.27	0.25
C_p	5	6.85
$Press_p$	2893.88	2943.18

(c): Recovered cases models

	AIC Model	BIC Model
p	13	5
SSE	302.65	466.12
MSE	3.06	4.35
R^2	0.78	0.66
R_a^2	0.75	0.65
C_p	-41.25	-33.08
$Press_p$	539	711.68

(d): Active cases models

Table 4: Comparison of $MSPE$ values between AIC and BIC models

	AIC Model	BIC Model
$MSPE_{valid}$	7.51	5.65
$MSPE_{train}$	1.32	1.66

(a): Confirmed cases models

	AIC Model	BIC Model
$MSPE_{valid}$	67.15	12.43
$MSPE_{train}$	7.51	33.11

(b): Death cases models

	AIC Model	BIC Model
$MSPE_{valid}$	10.25	8.93
$MSPE_{train}$	19.33	20.01

(c): Recovered cases models

	AIC Model	BIC Model
$MSPE_{valid}$	4.2	5.66
$MSPE_{train}$	2.7	4.16

(d): Active cases models

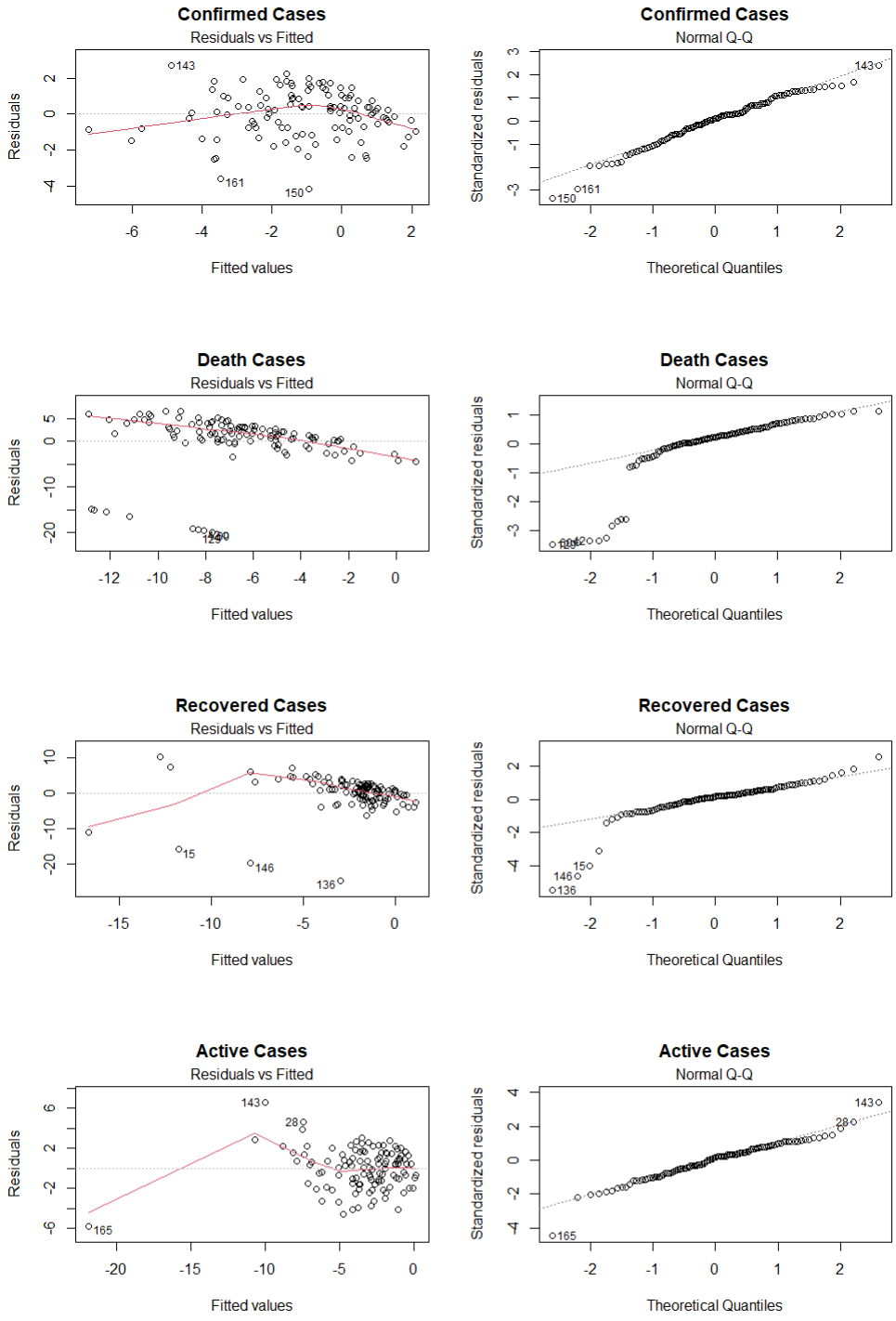


Figure 6: The plots on the left are the residuals vs fitted values plots for each response variable using the BIC models, whereas the plots on the right are their respective QQ plots. Bar a few possible outliers, it is obvious to see that the active cases and the confirmed cases models satisfy the normal error model assumptions.

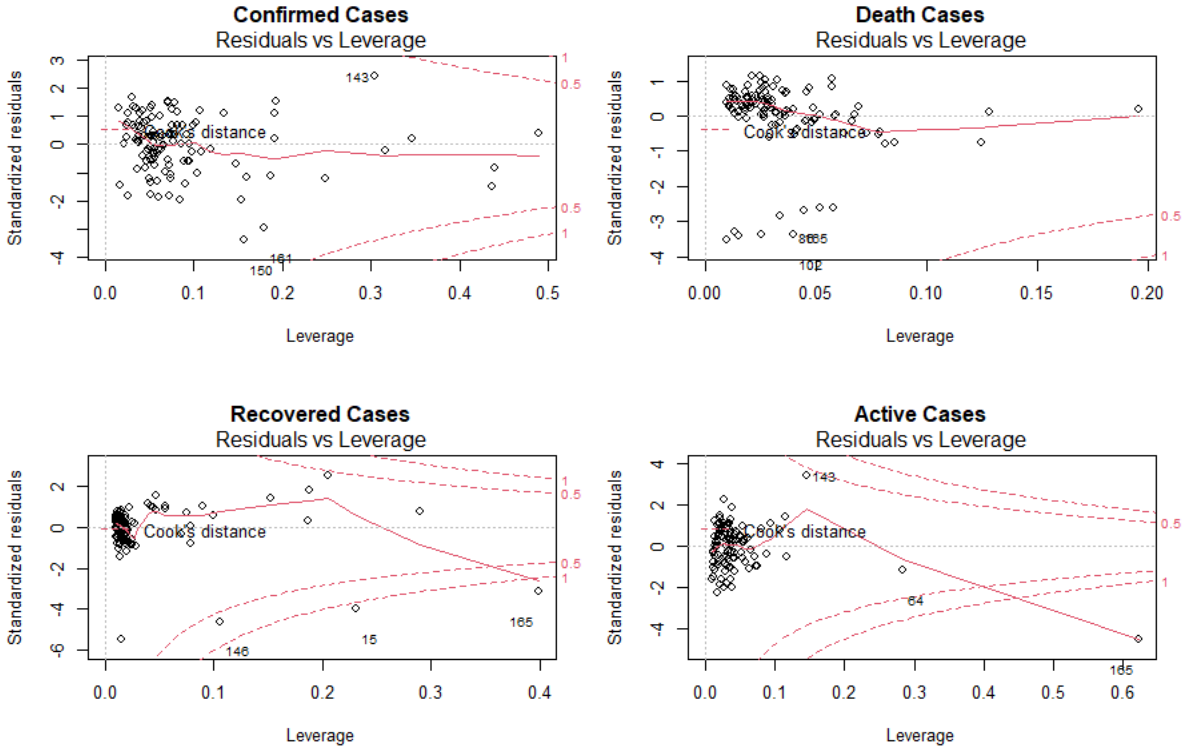
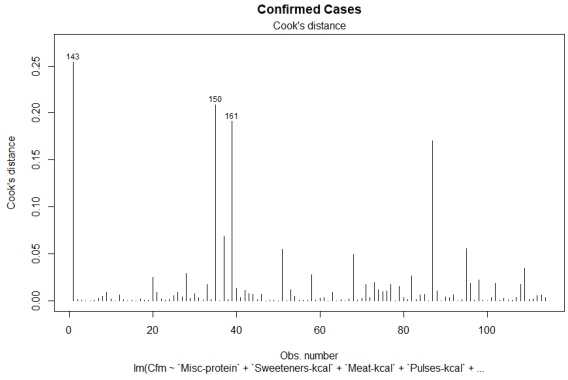
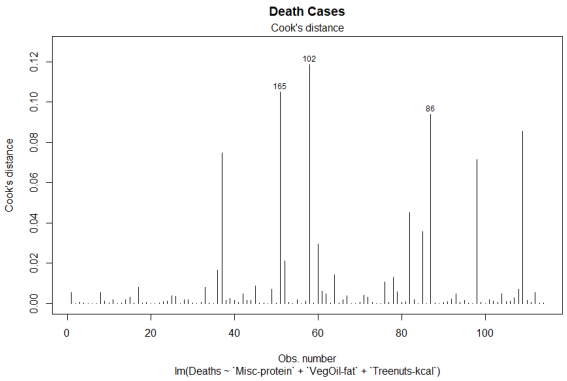


Figure 7: Residual vs Leverage plots for all the response variables



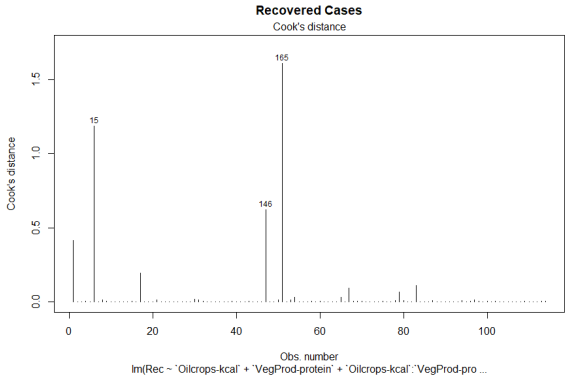
Index	Country	Cook's
143	Sri Lanka	0.25
150	Thailand	0.21
161	Tanzania	0.19
86	Laos	0.17

Figure 8: Significantly large values of Cook's distance in the confirmed cases model



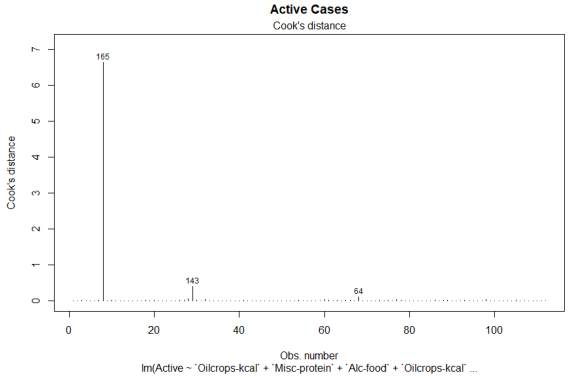
Index	Country	Cook's
102	Mongolia	0.118
165	Vanuatu	0.105
86	Loas	0.094
25	Combodia	0.086
42	Dominica	0.075
151	Timor-Leste	0.071

Figure 9: Significantly large values of Cook's distance in the death cases model



Index	Country	Cook's
165	Vanuatu	1.61
15	Belgium	1.19
146	Sweden	0.62
143	Sri Lanka	0.41

Figure 10: Significantly large values of Cook's distance in the recovered cases model



Index	Country	Cook's
165	Vanuatu	6.62
143	Sri Lanka	0.39

Figure 11: Significantly large values of Cook's distance in the active cases model

4.2 Appendix 2: R Codes

```
#####################################
# load all data
fat = read.table("Fat_Supply_Quantity_Data.csv", header = TRUE, sep = ",")
```

```r
kcal = read.table("Food_Supply_kcal_Data.csv", header = TRUE, sep = ",")
food = read.table("Food_Supply_Quantity_kg_Data.csv", header = TRUE, sep = ",")
protein = read.table("Protein_Supply_Quantity_Data.csv", header = TRUE, sep = ",")

####################################
# renaming variables
dfAsString = function(x){
  deparse(substitute(x))
}
colRename = function(x, name) {
  names(x) = c("Country", "Alc", "AnimalProd", "AnimalFats",
               "AqProd", "Cereal", "Eggs", "Seafood", "Fruits",
               "Meat", "Misc", "Milk", "Offals", "Oilcrops",
               "Pulses", "Spices", "StRt", "Stim", "SugarCrops",
               "Sweeteners", "Treenuts", "VegProd", "VegOil", "Veg",
               "Obesity", "Undernourished", "Cfm", "Deaths", "Rec",
               "Active", "Pop", "Unit")
  names(x)[2:24] = paste(names(x)[2:24], name, sep = "-")
  return(x)
}
fat = colRename(fat, dfAsString(fat))
kcal = colRename(kcal, dfAsString(kcal))
food = colRename(food, dfAsString(food))
protein = colRename(protein, dfAsString(protein))

####################################
## exploratory data analysis
# check correlation between measurements of same food categories
library(GGally)
for (i in 2:24){
  tempDf = as.data.frame(cbind(fat = fat[,i], kcal = kcal[,i], food = food[,i], protein =
    protein[,i]))
  title = strsplit(names(fat)[i], "-")[[1]]
  print(ggpairs(tempDf, title = title,
                upper = list(continuous = "points"), lower = list(continuous = "cor"),
                progress = FALSE))
  print(colSums(cor(tempDf)))
}

# remove unnecessary columns
redDf = cbind(fat[c(1,2,4,6,9,11,12,18,19,20,23)],
              kcal[c(3,5,6,7,8,10,12,14:17,20:24)],
              food[c(2:4,13,18,19,20,22,24)],
              protein[c(4,9,11:31)])
dim(redDf)

# remove X variables with unproper distribution
# quick view of histograms for all X variables
for (i in 2:52){
  hist(redDf[,i], xlab = names(redDf)[i], main = names(redDf)[i])
}

# remove unnecessary columns
drops = c("Alc-fat", "SugarCrops-fat", "Sweeteners-fat", "AqProd-kcal", "Sweeteners-food", "
    SugarCrops-protein")
redDf = redDf[,!(names(redDf) %in% drops)]

# remove X variables with extremely high VIF
VIF = diag(solve(cor(redDf[,2:46])))
ord = order(VIF, decreasing = TRUE)
VIF[ord]

# remove unnecessary columns
drops = c("AnimalFats-food", "Veg-food", "AnimalProd-kcal", "VegProd-kcal")
redDf = redDf[,!(names(redDf) %in% drops)]
```

```r
dim(redDf)

# checking correlation between Y variables
cor(na.omit(redDf)[,45:48])
dim(na.omit(redDf))

# best correlation between X and Y variables
bestCorDf = data.frame(Ord = 1:10)
for (i in 45:48){
  corrXY = cor(na.omit(redDf[,c(2:42,i)]))[,42]
  ord = order(abs(corrXY), decreasing = TRUE)
  bestCorDf = cbind(bestCorDf,
                    names(corrXY[ord[2:11]]),
                    corrXY[ord[2:11]])
}
names(bestCorDf) = c("Ord", "varCfm", "corCfm", "varD", "corD", "varRec", "corRec", "varAct"
    , "corAct")
row.names(bestCorDf) = NULL
bestCorDf

# distribution of Y variables
hist(redDf$Cfm, xlab = "Cfm", main = "Confirmed Cases")
hist(redDf$Death, xlab = "Death", main = "Death")
hist(redDf$Rec, xlab = "Rec", main = "Recovered")
hist(redDf$Active, xlab = "Active", main = "Active")

# box cox transformation
library(MASS)
fit = lm(Cfm ~ ., data = redDf[c(2:42, 45)]); boxcox(fit)
fit = lm(I(Deaths+.00001) ~ ., data = redDf[c(2:42, 46)]); boxcox(fit)
fit = lm(I(Rec+.00001) ~ ., data = redDf[c(2:42, 47)]); boxcox(fit)
fit = lm(I(Active+.00001) ~ ., data = redDf[c(2:42, 48)]); boxcox(fit)

# log transformation of Y variables
hist(log(fat$Cfm), xlab = "log(Cfm)", main = "log(Confirmed Cases)")
hist(log(fat$Death), xlab = "log(Death)", main = "log(Death)")
hist(log(fat$Rec), xlab = "log(Rec)", main = "log(Recovered)")
hist(log(fat$Active), xlab = "log(Active)", main = "log(Active)")

##########################################
## model fitting
# omit NA's and form 4 separate data frames
cfmDf = na.omit(redDf[,c(1:42, 45)])
deathDf = na.omit(redDf[,c(1:42,46)])
recDf = na.omit(redDf[,c(1:42,47)])
actDf = na.omit(redDf[,c(1:42,48)])

# check dimensions of each dataframe
dim(cfmDf)
dim(deathDf)
dim(recDf)
dim(actDf)

# transformation of Y variables
cfmDf$Cfm = log(cfmDf$Cfm)
deathDf$Deaths = log(deathDf$Deaths + 1e-12)
recDf$Rec = log(recDf$Rec + 1e-12)
actDf$Active = log(actDf$Active + 1e-12)

# split data into train and validation
set.seed(10)
ind1 = sample(1:dim(cfmDf)[1], size = 0.7*dim(cfmDf)[1])
ind2 = sample(1:dim(actDf)[1], size = 0.7*dim(actDf)[1])
# train
cfmDf.train = cfmDf[ind1,]
```

```r
deathDf.train = deathDf[ind1,]
recDf.train = recDf[ind1,]
actDf.train = actDf[ind2,]
#valid
cfmDf.valid = cfmDf[-ind1,]
deathDf.valid = deathDf[-ind1,]
recDf.valid = recDf[-ind1,]
actDf.valid = actDf[-ind2,]

# check distribution of response variables in train and validation sets
hist(cfmDf.train$Cfm, main = "Train")
hist(cfmDf.valid$Cfm, main = "Valid")
hist(deathDf.train$Deaths, main = "Train")
hist(deathDf.valid$Deaths, main = "Valid")
hist(recDf.train$Rec, main = "Train")
hist(recDf.valid$Rec, main = "Valid")
hist(actDf.train$Active, main = "Train")
hist(actDf.valid$Active, main = "Valid")

# model fitting for confirmed cases
cfm.fit1 = lm(Cfm ~ 1, data = cfmDf.train[,-1])

# fitting AIC model
cfm.fit2 = stepAIC(cfm.fit1,
                   scope = list(upper = lm(Cfm ~ ., data = cfmDf.train[,-1]),
                                lower = cfm.fit1),
                   direction = "both", trace = 0, k = 2)
summary(cfm.fit2)
cfm.fit4 = stepAIC(cfm.fit1,
                   scope = list(upper = lm(Cfm ~ (`Misc-protein` + `Sweeteners-kcal` + `Meat-kcal` +
                                                    `Pulses-kcal` + `Stim-protein` + `Oilcrops-kcal` + `Fruits-protein` +
                                                    `Eggs-kcal` + `Alc-food` + `Treenuts-kcal` + `Spices-kcal` +
                                                    `Sweeteners-protein`)^2,
                                            data = cfmDf.train[, -1]),
                                lower= cfm.fit1),
                   direction = "both", trace = 0, k = 2)
summary(cfm.fit4)
# fitting BIC model
cfm.fit3 = stepAIC(cfm.fit1,
                   scope = list(upper = lm(Cfm ~ .^2, data = cfmDf.train[,-1]),
                                lower = cfm.fit1),
                   direction = "both", trace = 0, k = log(nrow(cfmDf.train)))
summary(cfm.fit3)

# model fitting for death cases
death.fit1 = lm(Deaths ~ 1, data = deathDf.train[,-1])
# fitting AIC model
death.fit2 = stepAIC(death.fit1,
                     scope = list(upper = lm(Deaths ~ ., data = deathDf.train[,-1]),
                                  lower = death.fit1),
                     direction = "both", trace = 0, k = 2)
summary(death.fit2)
death.fit4 = stepAIC(death.fit1,
                     scope = list(upper = lm(formula = Deaths ~ (`Misc-protein` + `Treenuts-kcal` + `Meat-kcal` +
                                                                   `Eggs-kcal` + `Spices-kcal` + `Pulses-protein` + `Stim-protein` +
                                                                   `Oilcrops-kcal` + `Alc-food` + `Veg-protein` + `Veg-kcal` +
                                                                   `Misc-fat`)^2,
                                              data = deathDf.train[, -1]),
                                  lower = death.fit1),
```

```
                      direction = "both", trace = 0, k = 2)
summary(death.fit4)
# fitting BIC model
death.fit3 = stepAIC(death.fit1,
                      scope = list(upper = lm(Deaths ~ .^2, data = deathDf.train[,-1]),
                                   lower = death.fit1),
                      direction = "both", trace = 0, k = log(nrow(deathDf.train)))
summary(death.fit3)

# model fitting for recovery cases
rec.fit1 = lm(Rec ~ 1, data = recDf.train[,-1])
# fitting AIC model
rec.fit2 = stepAIC(rec.fit1,
                   scope = list(upper = lm(Rec ~ .^2, data = recDf.train[,-1]),
                                lower = rec.fit1),
                   direction = "both", trace = 0, k = 2)
summary(rec.fit2)
# fitting BIC model
rec.fit3 = stepAIC(rec.fit1,
                   scope = list(upper = lm(Rec ~ .^2, data = recDf.train[,-1]),
                                lower = rec.fit1),
                   direction = "both", trace = 0, k = log(nrow(recDf.train)))
summary(rec.fit3)

# model fitting for active cases
act.fit1 = lm(Active ~ 1, data = actDf.train[,-1])
# fitting AIC model
act.fit2 = stepAIC(act.fit1,
                   scope = list(upper = lm(Active ~ ., data = actDf.train[,-1]),
                                lower = act.fit1),
                   direction = "both", trace = 0, k = 2)
summary(act.fit2)
act.fit4 = stepAIC(act.fit1,
                   scope = list(upper = lm(formula = Active ~ (`Oilcrops-kcal` + `Misc-
    protein` + `Meat-kcal` +
                                                                `Treenuts-protein` + `
    Pulses-kcal` + `Alc-food` + `Sweeteners-protein`)^2,
                                           data = actDf.train[, -1]),
                                lower = act.fit1),
                   direction = "both", trace = 0, k = 2)
summary(act.fit4)
# fitting BIC model
act.fit3 = stepAIC(act.fit1,
                   scope = list(upper = lm(Active ~ .^2, data = actDf.train[,-1]),
                                lower = act.fit1),
                   direction = "both", trace = 0, k = log(nrow(actDf.train)))
summary(act.fit3)

##########################################################
## model diagnostics
# define functions
# compute sse, mse, R2, adjusted R2, Cp, Pressp statistics
modelDiag = function(obj, sigma2){
  n = length(obj$residuals)
  p = length(obj$coefficients)
  h = influence(obj)$hat
  sse = sum(obj$residuals^2)
  mse = sse/obj$df.residual
  r2 = summary(obj)$r.squared
  adjr2 = summary(obj)$adj.r.squared
  cp = sse/sigma2 - (n - 2*p)
  pressp = sum((obj$residuals/(1-h))^2)
  return (c(p = p, SSE = sse, MSE = mse, R2 = r2, Adj.R2 = adjr2, Cp = cp, PRESSp = pressp))
}

```

```
getMspe = function(fit, data){
  actual = data[,ncol(data)]
  pred = predict(fit, data)
  res = actual - pred
  m = nrow(data)
  return (c(MSPE = sum(res^2)/m))
}

modelValid = function(data, fit1, fit2){
  avg.sse1 = sum(fit1$residuals^2)/nrow(fit1$model)
  avg.sse2 = sum(fit2$residuals^2)/nrow(fit2$model)
  return (data.frame(AIC.valid = c(getMspe(fit1, data), `SSE/n` = avg.sse1),
                     BIC.valid = c(getMspe(fit2, data), `SSE/n` = avg.sse2)))
}

# internal and external validation for confirmed cases
# confirmed cases
cfm.fitfull = lm(formula = Cfm ~ `Misc-protein` + `Oilcrops-kcal` + `Stim-protein` +
                   `Fruits-protein` + `Pulses-kcal` + `Meat-kcal` + `Eggs-kcal` +
                   `Sweeteners-kcal` + `Alc-food` + `Treenuts-kcal` + `Oilcrops-kcal`:`Eggs-
    kcal` +
                   `Eggs-kcal`:`Sweeteners-kcal` + `Meat-kcal`:`Alc-food` +
                   `Fruits-protein`:`Pulses-kcal` + `Misc-protein`:`Alc-food`, data = cfmDf.
    train[,-1])
cfm.fit4.diag = modelDiag(cfm.fit4, sum(cfm.fitfull$residuals^2)/cfm.fitfull$df.residual) #
    train model
cfm.fit3.diag = modelDiag(cfm.fit3, sum(cfm.fitfull$residuals^2)/cfm.fitfull$df.residual)
cfm.diagComp = data.frame(AIC.train = cfm.fit4.diag, BIC.train = cfm.fit3.diag)
cfm.diagComp

cfm.diagComp.v = modelValid(cfmDf.valid[,-1], cfm.fit4, cfm.fit3) # valid model
cfm.diagComp.v

# internal and external validation for death cases
# death cases
death.fitfull = lm(Deaths ~ `Misc-protein` + `Meat-kcal` + `Eggs-kcal` +
                     `Spices-kcal` + `Pulses-protein` + `Stim-protein` + `Oilcrops-kcal` +
                     `Treenuts-kcal` + `Veg-protein` + `Misc-fat` + `Alc-food` +
                     `Eggs-kcal`:`Stim-protein` + `Eggs-kcal`:`Pulses-protein` +
                     `Meat-kcal`:`Pulses-protein` + `Meat-kcal`:`Treenuts-kcal` +
                     `Veg-protein`:`Misc-fat` + `Meat-kcal`:`Veg-protein` + `Misc-protein`:`
    Misc-fat` +
                     `Eggs-kcal`:`Oilcrops-kcal` + `Spices-kcal`:`Treenuts-kcal` +
                     `Meat-kcal`:`Misc-fat` + `Meat-kcal`:`Alc-food` + `Treenuts-kcal`:`Alc-
    food` +
                     `Eggs-kcal`:`Veg-protein` + `Pulses-protein`:`Oilcrops-kcal` +
                     `Pulses-protein`:`Treenuts-kcal` + `Misc-fat`:`Alc-food` +
                     `Veg-protein`:`Alc-food` + `Misc-protein`:`Oilcrops-kcal` +
                     `Stim-protein`:`Alc-food` + `Pulses-protein`:`Misc-fat` +
                     `Stim-protein`:`Veg-protein` + `Eggs-kcal`:`Misc-fat` + `VegOil-fat`,
    deathDf.train[, -1])
death.fit4.diag = modelDiag(death.fit4, sum(death.fitfull$residuals^2)/death.fitfull$df.
    residual) # train model
death.fit3.diag = modelDiag(death.fit3, sum(death.fitfull$residuals^2)/death.fitfull$df.
    residual)
death.diagComp = data.frame(AIC.train = death.fit4.diag, BIC.train = death.fit3.diag)
death.diagComp

death.diagComp.v = modelValid(deathDf.valid[,-1], death.fit4, death.fit3) # valid model
death.diagComp.v

# internal and external validation for recovered cases
# recovery cases
rec.fitfull = rec.fit2
rec.fit2.diag = modelDiag(rec.fit2, sum(rec.fitfull$residuals^2)/rec.fitfull$df.residual) #
```

```
    train model
rec.fit3.diag = modelDiag(rec.fit3, sum(rec.fitfull$residuals^2)/rec.fitfull$df.residual)
rec.diagComp = data.frame(AIC.train = rec.fit2.diag, BIC.train = rec.fit3.diag)
rec.diagComp

rec.diagComp.v = modelValid(recDf.valid[,-1], rec.fit2, rec.fit3) # valid model
rec.diagComp.v

# internal and external validation for activ cases
# active cases
act.fitfull = lm(formula = Active ~ `Misc-protein` + `Offals-protein` + `Meat-kcal` +
                   `Eggs-kcal` + `Spices-protein` + `Stim-fat` + `Meat-kcal`:`Spices-protein
    ` +
                   `Misc-protein`:`Stim-fat` + `Misc-protein`:`Meat-kcal` + `Offals-food`,
                 data = actDf.train[, -1])
act.fit4.diag = modelDiag(act.fit4, sum(act.fitfull$residuals^2)/act.fitfull$df.residual) #
    train model
act.fit3.diag = modelDiag(act.fit3, sum(act.fitfull$residuals^2)/act.fitfull$df.residual)
act.diagComp = data.frame(AIC.train = act.fit4.diag, BIC.train = act.fit3.diag)
act.diagComp

act.diagComp.v = modelValid(actDf.valid[,-1], act.fit4, act.fit3) # valid model
act.diagComp.v

## checking for outlying cases
# function to calculate Cook's distance
getCooksDist = function(obj){
  e = obj$residuals
  p = length(obj$coefficients)
  mse = sum(e^2)/obj$df.residual
  h = influence(obj)$hat
  return (c(e^2*h/(p*mse*(1-h)^2)))
}

# model diagnostics for confirmed cases
# residual analysis
plot(cfm.fit4, which = 1:2, main = "AIC Model")
plot(cfm.fit3, which = 1:2, main = "BIC Model")
# outlying cases
plot(cfm.fit4, which = 4:6)
plot(cfm.fit3, which = 4:6)
# calculate cook's distance and determine outliers
cooks = getCooksDist(cfm.fit3)
cfmOutliers = cbind(cfmDf.train[which(cooks > 4/(cfm.fit3$df.residual)),c(1,43)], CooksDist
    = cooks[which(cooks > 4/(cfm.fit3$df.residual))])
cfmOutliers[order(cfmOutliers$CooksDist, decreasing = TRUE),]
# examine influence of possible outlier
cfm.fit3.rmOutlier = lm(formula = cfm.fit3$call$formula, data = cfmDf.train[-c
    (143,150,161,86),])
per.change = abs((cfm.fit3$fitted.values - predict.lm(cfm.fit3.rmOutlier, cfmDf.train))/cfm.
    fit3$fitted.values)*100
summary(per.change)
# remove outliers
cfmDf.train.rmOutlier = cfmDf.train[-c(143,150,161,86),]

# model diagnostics for death cases
# residual analysis
plot(death.fit4, which = 1:2, main = "AIC Model")
plot(death.fit3, which = 1:2, main = "BIC Model")
# outlying cases
plot(death.fit4, which = 4:6)
plot(death.fit3, which = 4:6)
# calculate cook's distance and determine outliers
cooks = getCooksDist(death.fit3)
deathOutliers = cbind(deathDf.train[which(cooks > 4/(death.fit3$df.residual)),c(1,43)],
```

```
    CooksDist = cooks[which(cooks > 4/(death.fit3$df.residual))])
deathOutliers[order(deathOutliers$CooksDist, decreasing = TRUE),]
# examine influence of possible outlier
death.fit3.rmOutlier = lm(formula = death.fit3$call$formula, data = deathDf.train[-c
    (102,165,86,25,42,151),])
per.change = abs((death.fit3$fitted.values - predict.lm(death.fit3.rmOutlier, deathDf.train)
    )/death.fit3$fitted.values)*100
summary(per.change)
# remove outlier
deathDf.train.rmOutlier = deathDf.train[-c(102,165,86,25,42,151),]

# model diagnostics for recovered cases
# residual analysis
plot(rec.fit2, which = 1:2, main = "AIC Model")
plot(rec.fit3, which = 1:2, main = "BIC Model")
# outlying cases
plot(rec.fit2, which = 4:6)
plot(rec.fit3, which = 4:6)
# calculate cook's distance and determine outliers
cooks = getCooksDist(rec.fit3)
recOutliers = cbind(recDf.train[which(cooks > 4/(rec.fit3$df.residual)),c(1,43)], CooksDist
    = cooks[which(cooks > 4/(rec.fit3$df.residual))])
recOutliers[order(recOutliers$CooksDist, decreasing = TRUE),]
# examine influence of possible outlier
rec.fit3.rmOutlier = lm(formula = rec.fit3$call$formula, data = recDf.train[-c
    (165,15,146,143),])
per.change = abs((rec.fit3$fitted.values - predict.lm(rec.fit3.rmOutlier, recDf.train))/rec.
    fit3$fitted.values)*100
summary(per.change)
# remove outliers from dataset
recDf.train.rmOutlier = recDf.train[-c(165,15,146,143),]

# model diagnostics for active cases
# residual analysis
plot(act.fit4, which = 1:2, main = "AIC Model")
plot(act.fit3, which = 1:2, main = "BIC Model")
# outlying cases
plot(act.fit4, which = 4:6)
plot(act.fit3, which = 4:6)
# calculate cook's distance and determine outliers
cooks = getCooksDist(act.fit3)
actOutliers = cbind(actDf.train[which(cooks > 4/(act.fit3$df.residual)),c(1,43)], CooksDist
    = cooks[which(cooks > 4/(act.fit3$df.residual))])
actOutliers[order(actOutliers$CooksDist, decreasing = TRUE),]
# examine influence of possible outlier
act.fit3.rmOutlier = lm(formula = act.fit3$call$formula, data = actDf.train[-c(165,143),])
per.change = abs((act.fit3$fitted.values - predict.lm(act.fit3.rmOutlier, actDf.train))/act.
    fit3$fitted.values)*100
summary(per.change)

##############################################################################
## final models: refit models without outliers with BIC criterion
# confirmed cases
cfmDf.rmOutlier = rbind(cfmDf.train.rmOutlier, cfmDf.valid)
cfm.fit.final = lm(cfm.fit3$call$formula, data = cfmDf.rmOutlier)
summary(cfm.fit.final)
anova(cfm.fit.final)

# death cases
deathDf.rmOutlier = rbind(deathDf.train.rmOutlier, deathDf.valid)
death.fit.final = lm(death.fit3$call$formula, data = deathDf.rmOutlier)
summary(death.fit.final)
anova(death.fit.final)

# recovered cases
```

```r
recDf.rmOutlier = rbind(recDf.train.rmOutlier, recDf.valid)
rec.fit.final = lm(rec.fit3$call$formula, data = recDf.rmOutlier)
summary(rec.fit.final)
anova(rec.fit.final)

# active cases
actDf.rmOutlier = rbind(actDf.train, actDf.valid)
act.fit.final = lm(act.fit4$call$formula, data = actDf.rmOutlier)
summary(act.fit.final)
anova(act.fit.final)
```