COMP3111H Task B3 Problem Statement, Solution Statement and Supplementary Notes

Chow, Hau Cheung Jasper (hcjchow / 20589533)

November 19, 2021

Problem Statement

In combating the spread of any epidemic, it is crucial to understand the factors that influence the number of COVID-related deaths. However, since population sizes in countries vary, it is better to analyse the standardised value; **total COVID-related deaths per million**. We want to answer the question: are there specific factors which can contribute (either positively or negatively) to the total COVID-related deaths per million, what are they, and to which extent? How much of the total COVID-related deaths per million can be explained by these variables? Can we predict the number of COVID deaths per million, and use that predictive model to shed light on which things governments should address to most effectively stem the virus' rampage?

Solution Statement

Most of these questions can be answered with the use of multiple linear regression. Linear regression is the technique of attempting to find a linear relationship between a response variable Y (the variable we want to predict) to a set of p predictor variables $\{X_1, ..., X_p\}$. Assuming we have n data points $(X_{11}, X_{12}, ..., X_{1p}, y_1), ...(X_{n1}, X_{n2}, ..., X_{np}, y_n)$, this is done by constructing an equation in the form $y_i = \beta_0 + \sum_{j=1}^{p}(\beta_j x_{ij}) + \epsilon_i$ where $\epsilon \sim N(0, \sigma^2)$ denotes the random error and is assumed to be independent. Since we don't know the value of the β_j coefficients, we need to estimate them.

The datapoints used are the values of each country at their most recent reporting date, for a total of $n = 231$ datapoints.

Linear regression methods are typically solved by ordinary least squares (OLS) estimates. Intuitively, the 'best-fit' equation is calculated by making the sum of all the predicted points $\hat{y_i}$ be as close to the actual points y_i. This is done by minimising $\sum_{i=1}^{n}(y_i - \hat{y_i})^2$, i.e. $\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}(\beta_j x_{ij}))^2$. The general form of the solution is given by $\hat{\beta} = (X^T X)^{-1} X^T y$ **(Equation 1)** where X is the data matrix, $y = (y_1, ..., y_n)$ is the vector of response variables, and $\hat{\beta} = (\hat{\beta_0}, \hat{\beta_1}, ...\hat{\beta_p})$ denotes our ESTIMATE of the coefficients.

How to Use

The given UI (navigate to the Task B3 tab) allows us to select any set of predictor variables from the checklist in the top left to conduct **regression analysis** on. (The response variable is always total number of COVID deaths per million.) Suppose we choose `life_expectancy` and `stringency_index` from the checklist box. Then we will be attempting to find $(\beta_0, \beta_1, \beta_2)$ such that $totaldeathspermillion = \beta_0 + \beta_1 lifeexpectancy + \beta_2 stringencyindex$ is the most consistent with the observed values of `total_deaths_per_million`.

Click "Generate Report." The console will report that the linear regression output has been calculated. You will see that the 4 buttons below (Regression Output, Residual Plot, Collinearity Analysis and Correlation Matrix) have been activated - these provide some simple tools for conducting regression analysis on the dataset, and in fact are flexible enough to work with additional variables if they are added as columns to `COVID_Dataset_v1.0.csv`.

In some cases, the report may not be generated. This is because the program will remove each datapoint which has even a single missing value for the chosen set of predictors. If the report is generated, however, a list of how many missing values were removed from each variable and the total number of datapoints removed will be written to the console.

If you would like to rerun the regression results for a different set of variables, simply select that set of variables from the dropdown checklist and click the "Generate Report" button again. Then click Regression Output, Residual Plot, Collinearity Analysis and Correlation Matrix buttons as needed.

Implementation Details

Task B3 was implemented with the help of Renjin, an interpreter for the R programming language written in Java. Renjin commands are executed on the JVM, same as any other Java code, and allows for two-way access between Java-declared variables and functions and R-declared variables and functions. R commands are easily executed with `engine.eval("some command here")` and the outputs of R expressions can instead be stored in the Renjin `SEXP` data type, which can be converted to Vectors of Double, Integer, etc.

Renjin was chosen to negate the need for implementing complex mathematical equations (such as the linear regression closed-form solution (**Equation 1**)) in Java.

Below is the explanation and rationale for each of the buttons on the UI:

1. Regression Output

Clicking this button lists the regression coefficients of each variable selected in the dropdown checklist in a table, and their associated p-values. A low p-value for a particular coefficient $\hat{\beta}_j$ (less than 0.05) suggests that the coefficient is significant (namely, the hypothesis test $H_0 : \beta_j = 0$ is rejected) and that the variable X_j should be included in the model.

Additionally, the multiple R-squared, adjusted R-squared, F-statistic and residual standard error $\hat{\sigma}$ will be shown. The R-squared values indicate how much of the variability in the response variable is explained by the chosen set of predictor variables (eg a multiple R-squared of 0.5 suggests 50% of the variability in `total_deaths_per_million` is explained by the set of chosen variables.) The adjusted R-squared adjusts this value to account for the fact that more variables always explains at least the same amount of variability, but the inclusion of additional variables may be irrelevant. In general, the higher the adjusted R-squared of a model (closer to 1), the better. The F-statistic is used to judge how good the model is compared to the intercept-only model (no predictors) i.e. $Y = \beta_0 + \epsilon$. If the F-statistic is low then that indicates that NONE of the selected variables were useful in predicting the total deaths per million, and if it is high then at least one predictor is useful in predicting the total deaths per million. A low F-statistic indicates a different set of predictors should be used. The residual standard error $\hat{\sigma}$ describes an unbiased estimate for the error variance, and is given as follows $\hat{\sigma} = \sqrt{SSE/n - p - 1}$ where p is the number of predictor variables. A high value of $\hat{\sigma}$ suggests that our predictions using this model may not be suitable.

Each row/coefficient of the regression output will be coloured from yellow to green, which yellow indicating insignificant coefficient (high p-values) and green indicating more significant coefficient (lower p-values).

However, linear regression is predicated on a set of assumptions, which if violated, can weaken the results of our conclusion or in the worst case, completely invalidate our analysis.

2. Residual Plot

One of the most important assumptions in regression analysis is the assumption of constant error variance $\epsilon_i \sim N(0, \sigma^2)$. When this assumption is violated, this model's predictive power is weakened, as the values of the predictors may influence the final result. Additionally, it may lead to us erroneously concluding that certain regression coefficients are significant when in fact they are not, resulting in keeping unnecessary variables in the model.

This assumption can be checked by examining the residual plots, i.e. we obtain the values of $e_i = y_i - \hat{y}_i$, standardise them to get r_i, and plot them against the fitted values $\hat{y}_i$. If there is a trend, such as a quadratic or linear trend, that suggests that the variance of the error term is not independent of the fitted values. This suggests that some variable transformation on some of the predictors may be necessary to correct the residuals.

3. Collinearity Analysis

The closed-form solution to least squares regression (**Equation 1**) is dependent on the assumption that the matrix $X^T X$ is invertible, which is violated if the columns of $X^T X$ are linearly dependent. Linear dependence implies that some predictors are unnecessary as the information they contain is also given by other variables. Without this assumption, the least squares solution would not be unique and the regression coefficients would lose their meaning, so we would not be able to quantify how much some variables affect the total deaths per million compared to others. This would also make prediction meaningless as the estimated values of the coefficients are sensitive to slight changes in values of the predictor variables.

Example of multicollinearity: Consider if we kept the number of individuals in the ages of 0-24, 25-29, 30-34, 35-39, ..., 80+ and also had a variable containing the total population size. Clearly, we could remove one of the groups like 0-24, since the number of people in that group is just the total population minus the sum of the number of people in every other age group.

One way of quantifying linear dependency is to use the Variance Inflation Factor (VIF), defined as follows, $VIF_i = \dfrac{1}{1 - R^2(X_i)}$ where $R^2(X_i)$ denotes the multiple R^2 of predictor X_i against all other predictors (i.e. the proportion of variability of predictor X_i that could be explained by the other predictors.) In general, $VIF_i \geq 10$ indicates multicollinearity and we should remove predictor i.

The VIFs will be coloured from red to green, which red indicating high VIFs (over 10), orange-yellow for VIFs in the 4-10 range, and green for VIFs close to 1.

Note that the VIF is clearly not defined if only one predictor is used. In this case, clicking the Collinearity Analysis button will simply inform the user of this via the console and not display any output.

4. Correlation Matrix

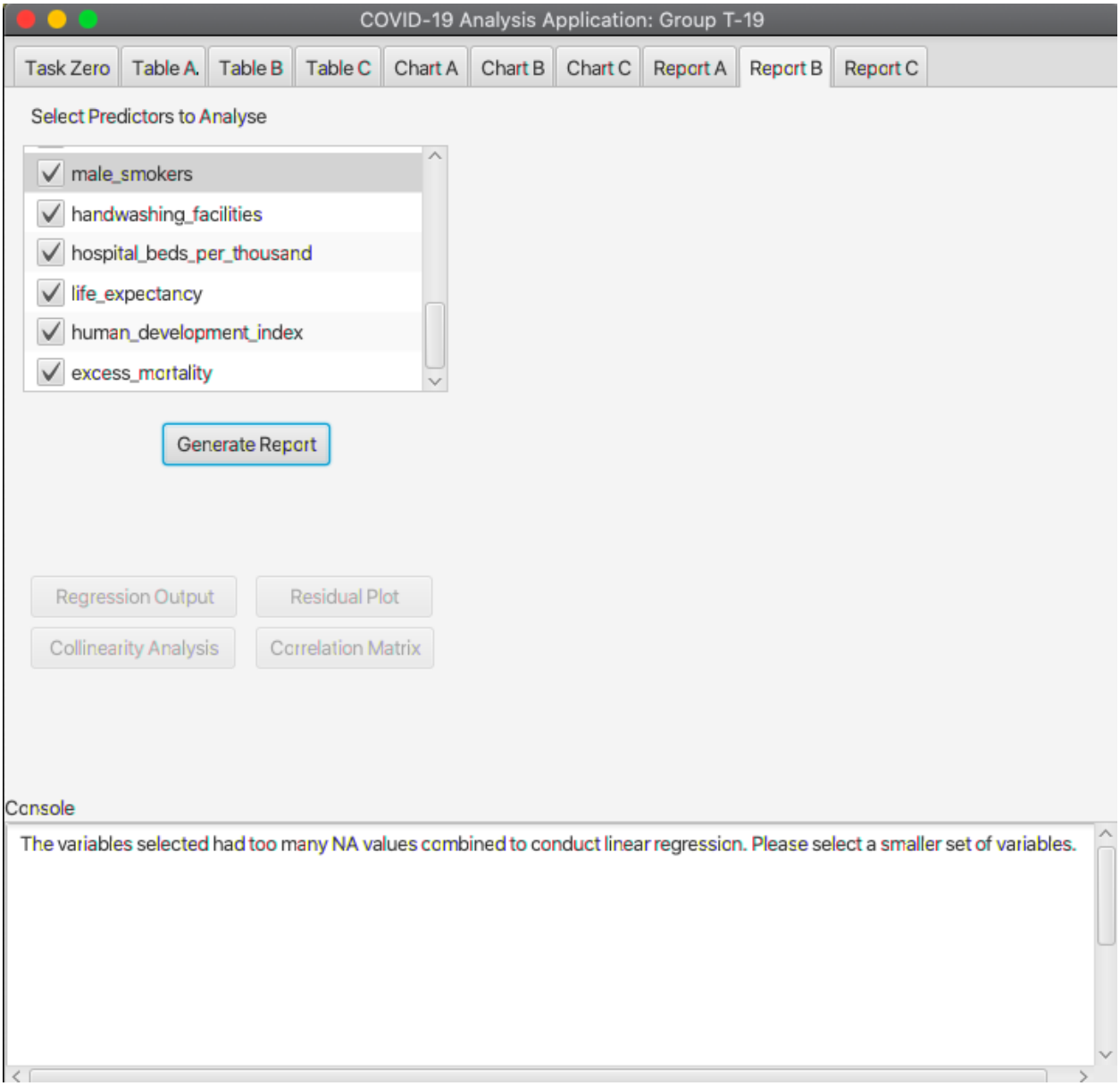
When conducting regression analysis, we may have to compare the effectiveness of different models. How should we compare Model A and Model B, for instance, if both have similar adjusted R-squared values? In variable selection, statisticians typically use the **forward selection procedure** to compare models. To do so, they start with the intercept-only model, and add the predictor with the highest absolute correlation value. If the predictor has a significant coefficient (low p-value) the process repeats and another predictor is selected.

The correlation matrix may also be used to determine which variables to drop if one or more of the VIFs is over 10, as a high correlation between two variables with high VIFs indicates linear dependency.

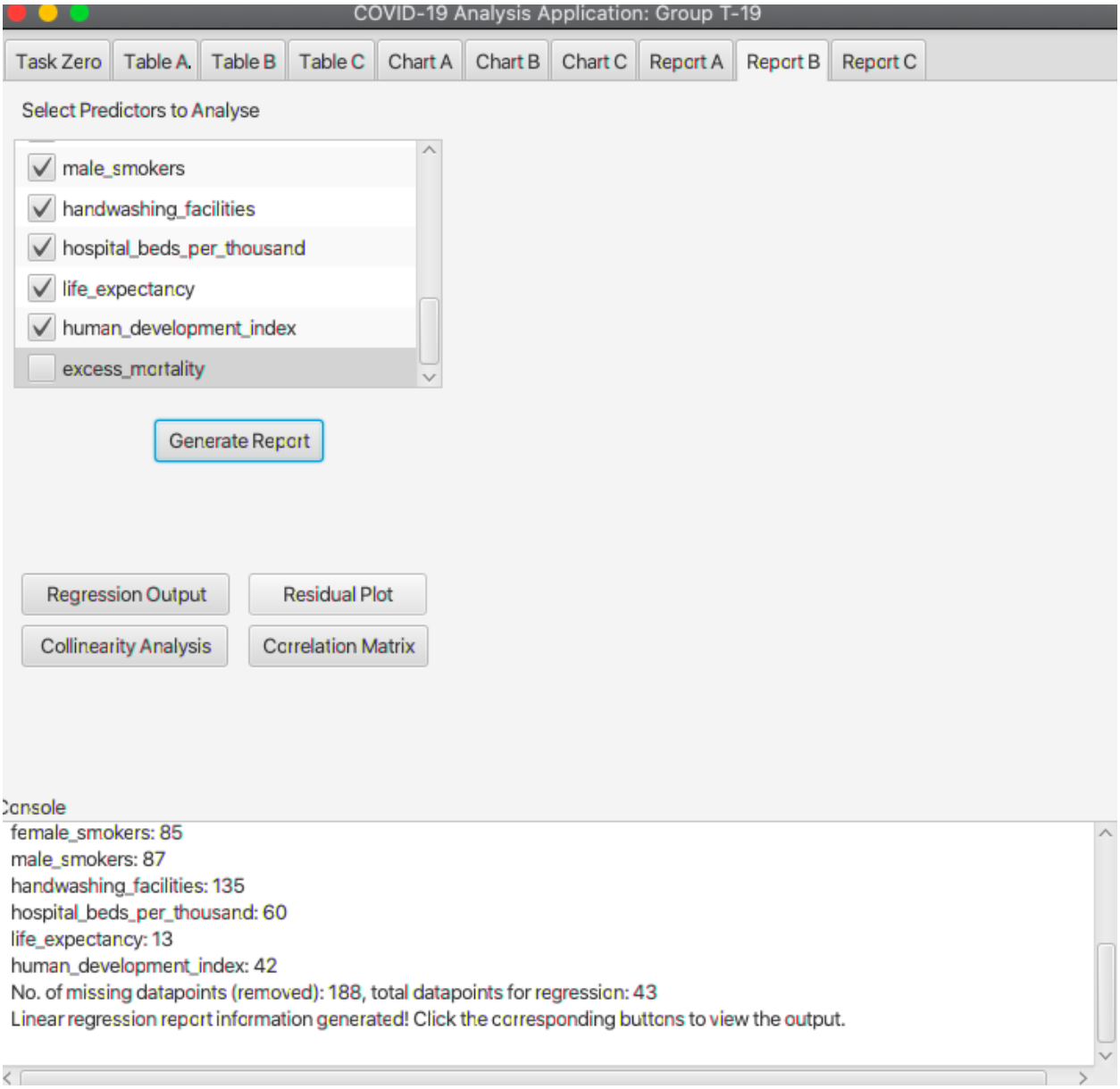
Each cell in the correlation matrix will also be coloured from red to green, which green indicating high positive correlation, red indicating high negative correlation, and yellow-orange indicating 0 correlation.

Sample usage of the program

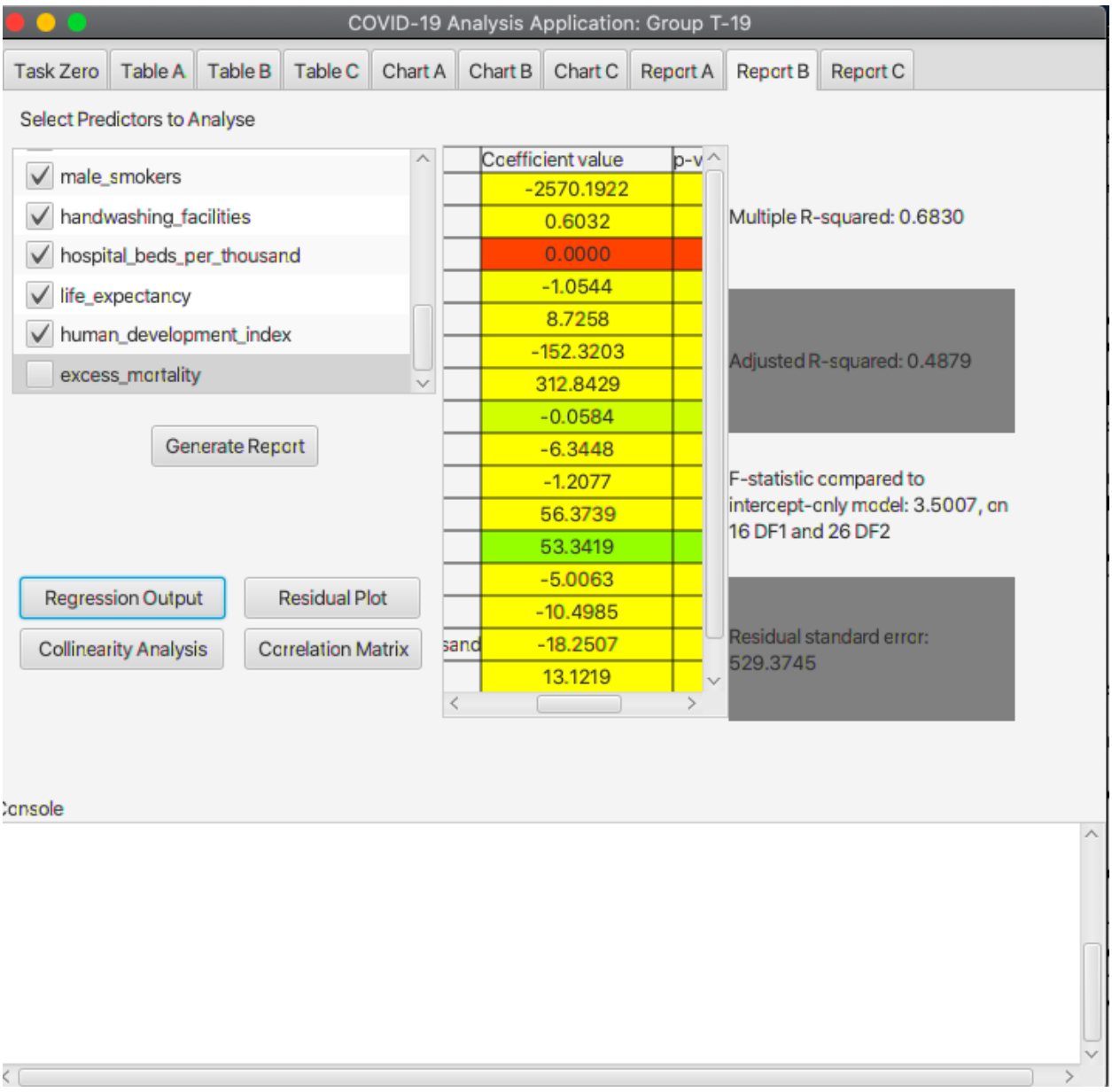
Here is some sample usage of the program. Let us try selecting all the variables at first.



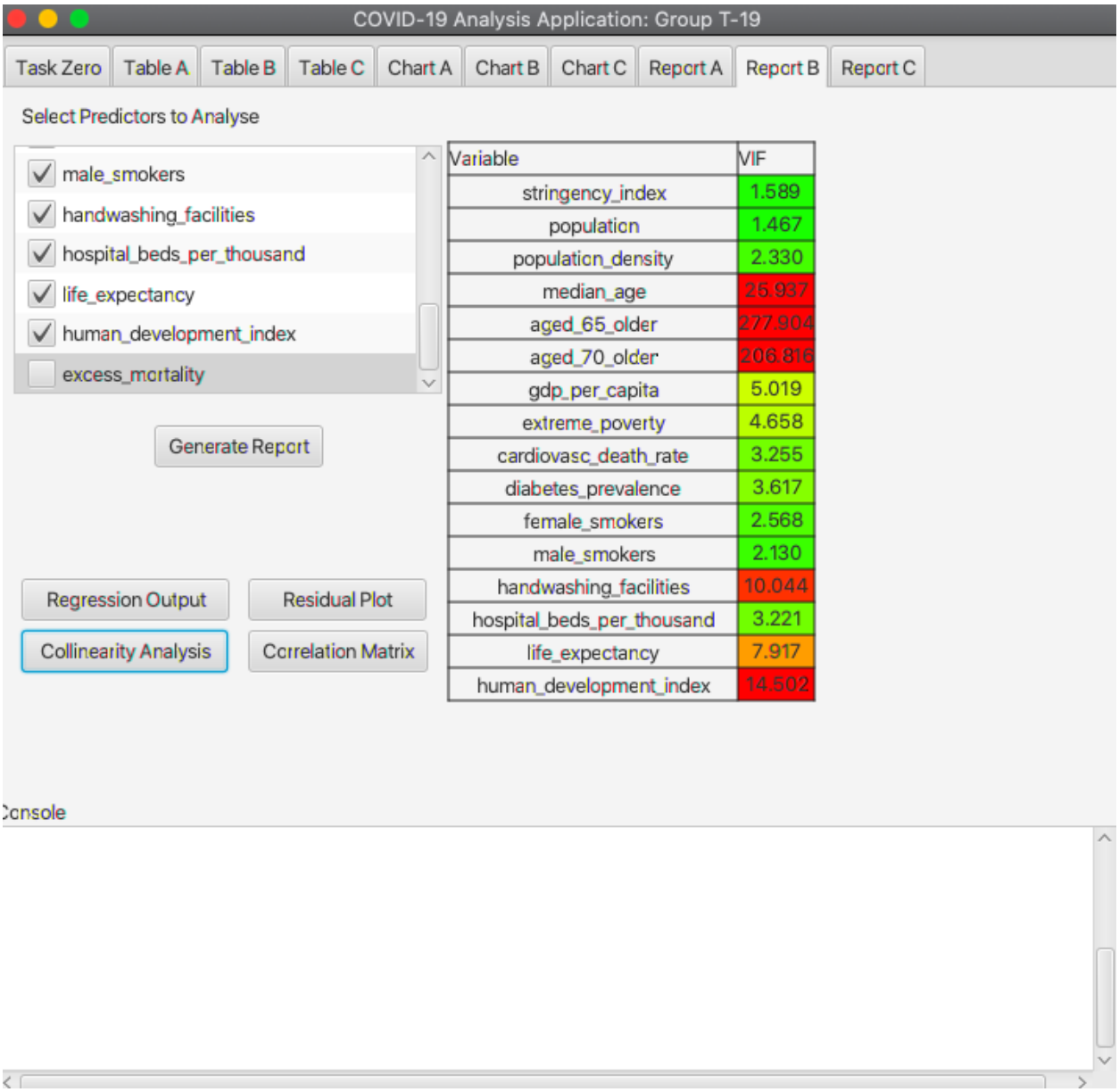
Clearly there are too many NA values! Let's start by removing some variables.



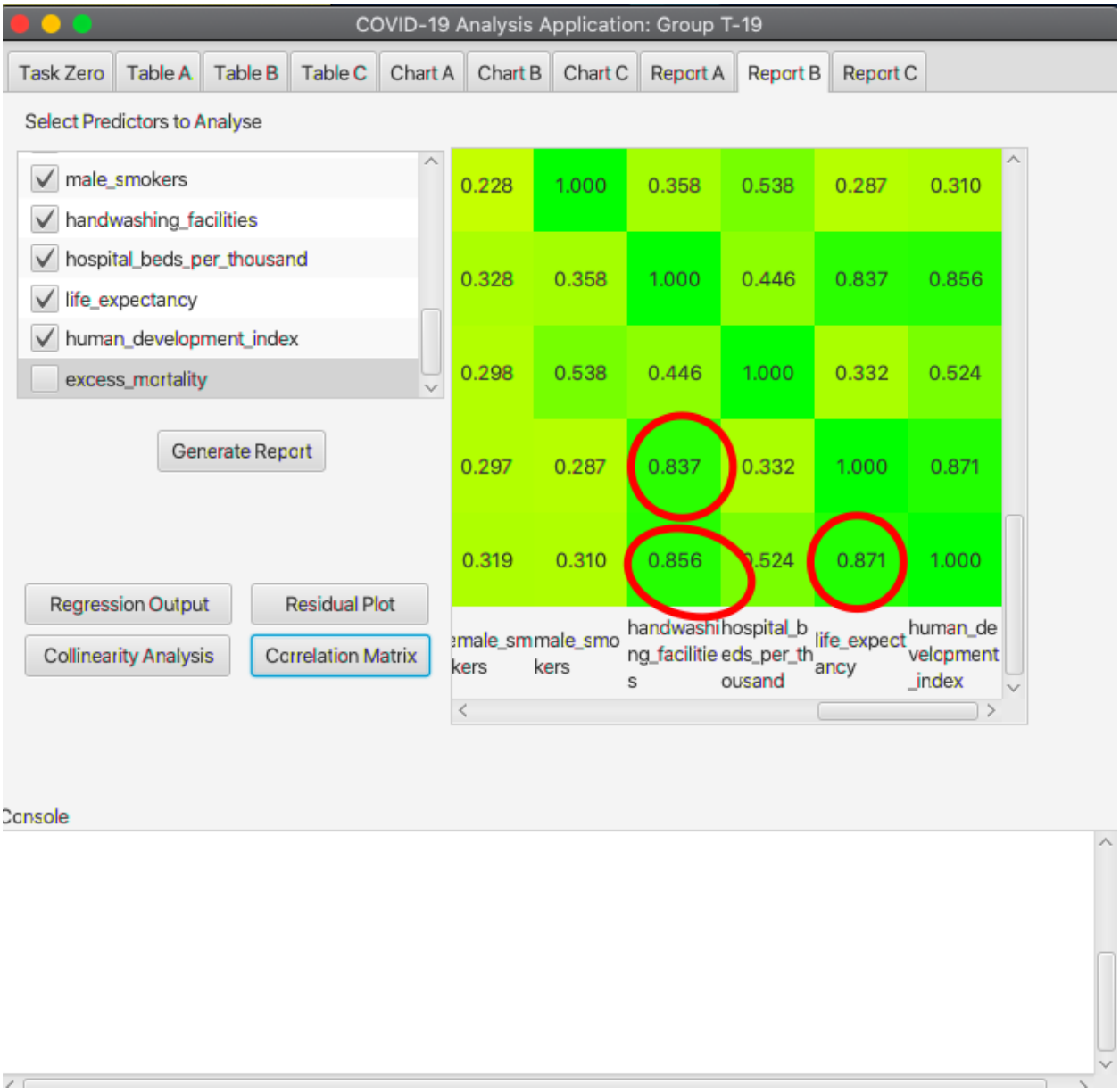
This produces a regression output as seen here:

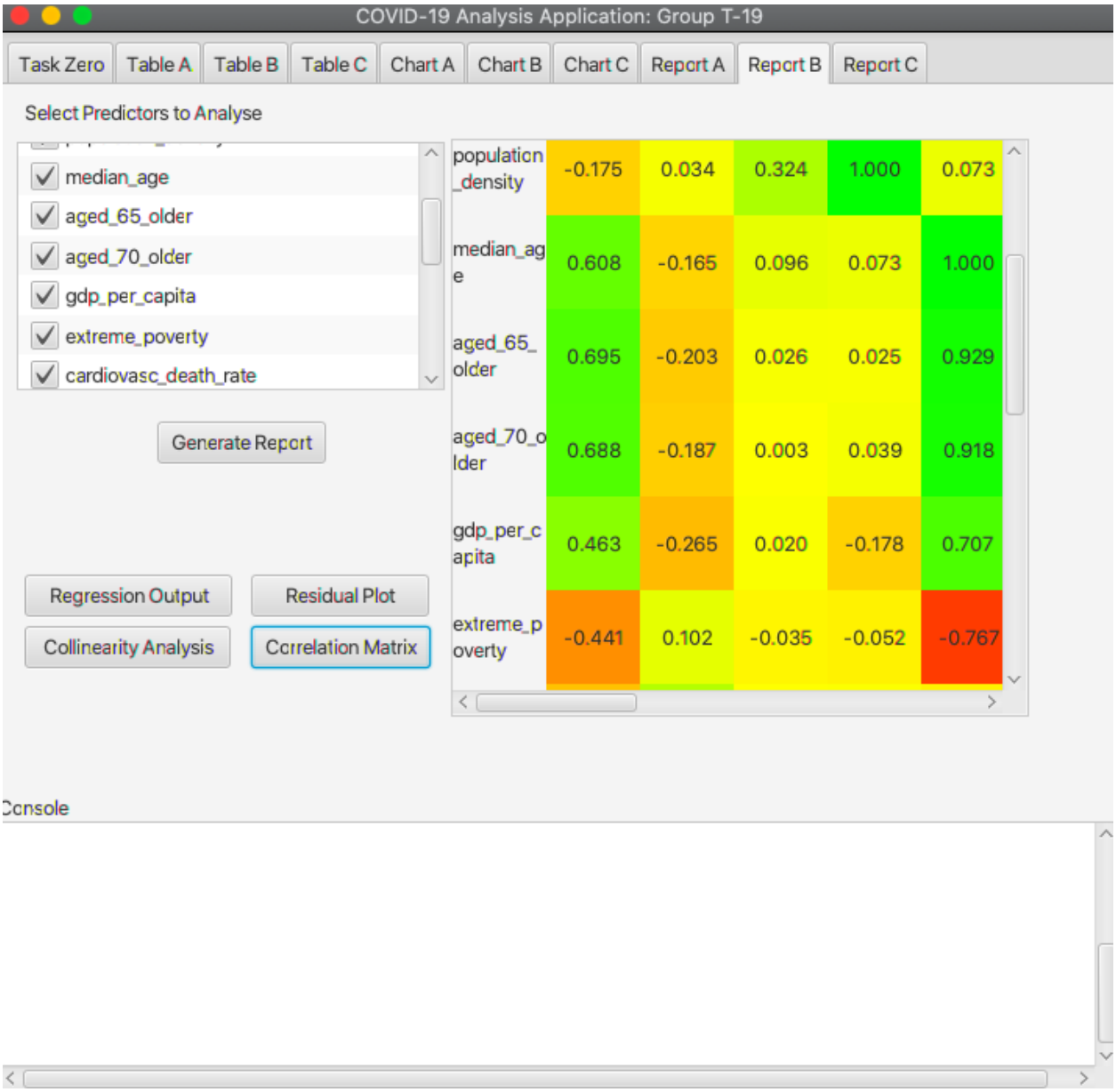


But we can see from the Collinearity Analysis that some of the VIFs are 10 or higher!

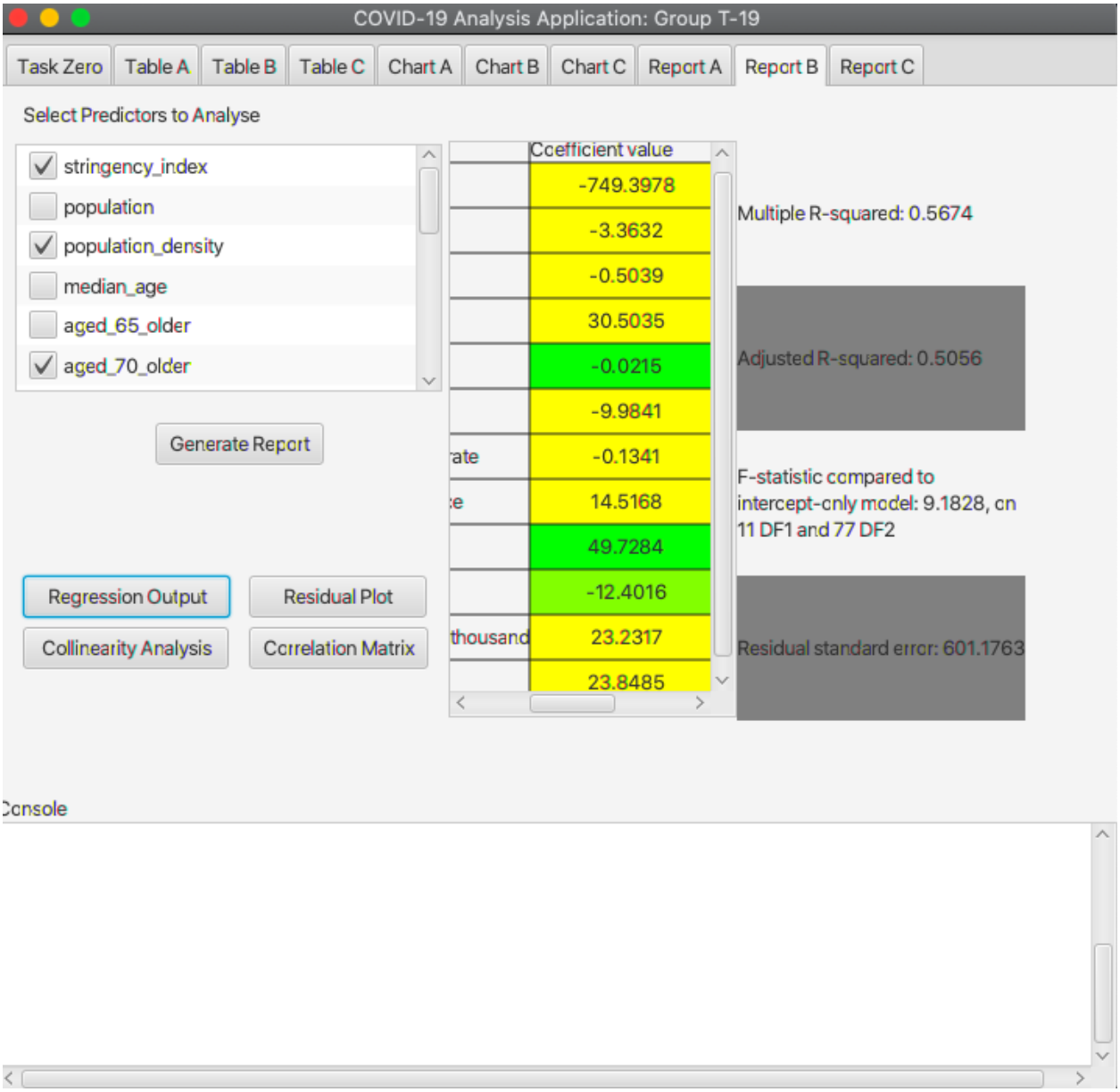


We should remove some of them. But which ones to remove? Let's examine the Correlation Matrix.

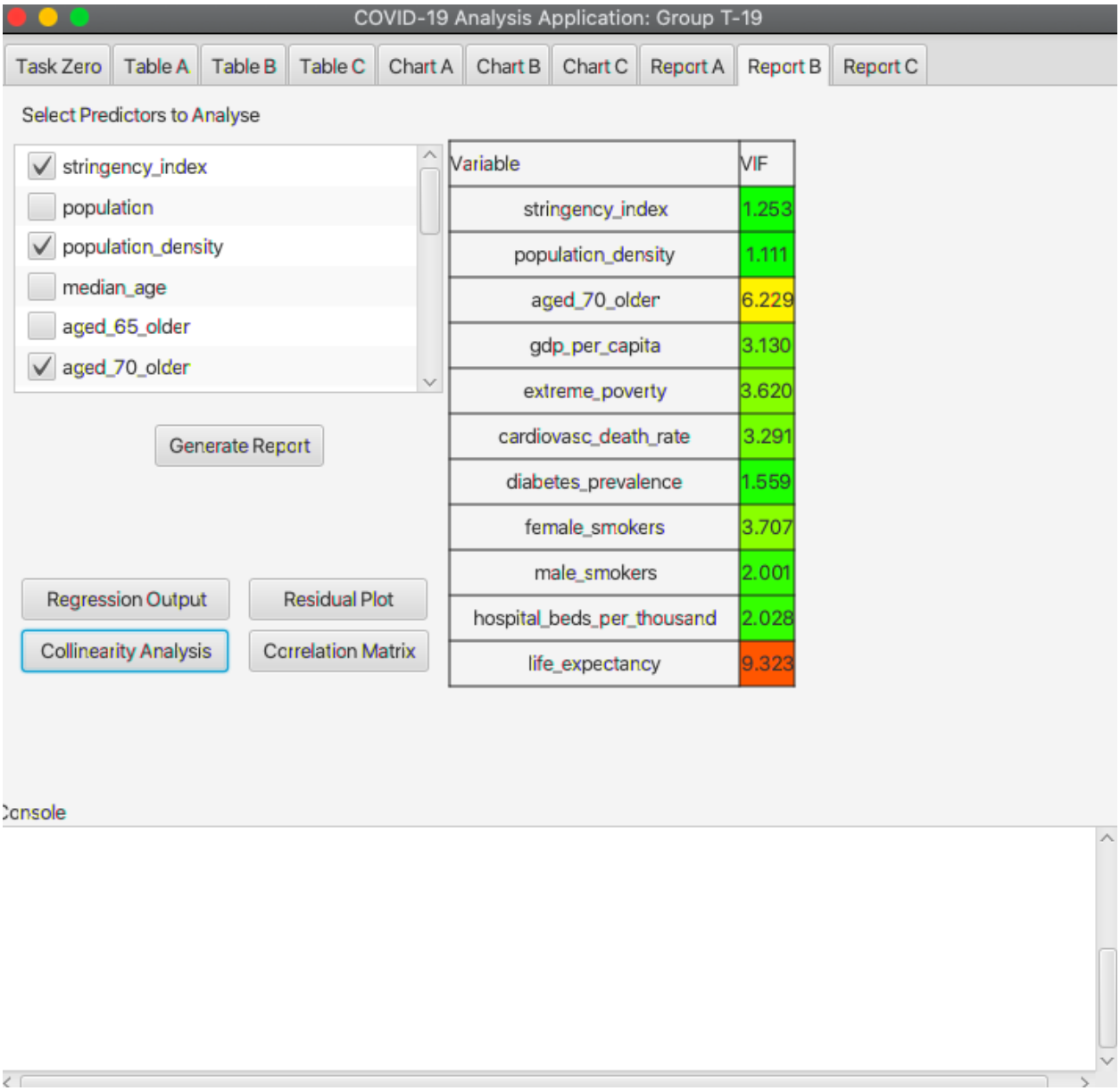

COVID-19 Analysis Application: Group T-19
Task Zero
Table A
Table B
Table C
Chart A
Chart B
Chart C
Report A
Report B
Report C
Select Predictors to Analyse
male_smokers
handwashing_facilities
hospital_beds_per_thousand
life_expectancy
human_development_index
excess_mortality
Generate Report
Regression Output
Residual Plot
Collinearity Analysis
Correlation Matrix
0.228 1.000 0.358 0.538 0.287 0.310
0.328 0.358 1.000 0.446 0.837 0.856
0.298 0.538 0.446 1.000 0.332 0.524
0.297 0.287 0.837 0.332 1.000 0.871
0.319 0.310 0.856 0.524 0.871 1.000
male_smokers
male_smokers
handwashing_facilities
hospital_beds_per_thousand
life_expectancy
human_development_index
Console



From the circled values, we see that `human_development_index` has high positive correlation with `handwashing_facilities` and `life_expectancy`. Also notice that `median_age`, `aged_65_older` and `aged_70_older` all have extremely high positive pairwise correlations. So let's say we choose to remove `median_age`, `aged_65_older`, `handwashing_facilities`, and `human_development_index` for now since those ones have VIF over 10. We also remove the `population` variable from the regression output since the p-value is extremely high, meaning it is insignificant.

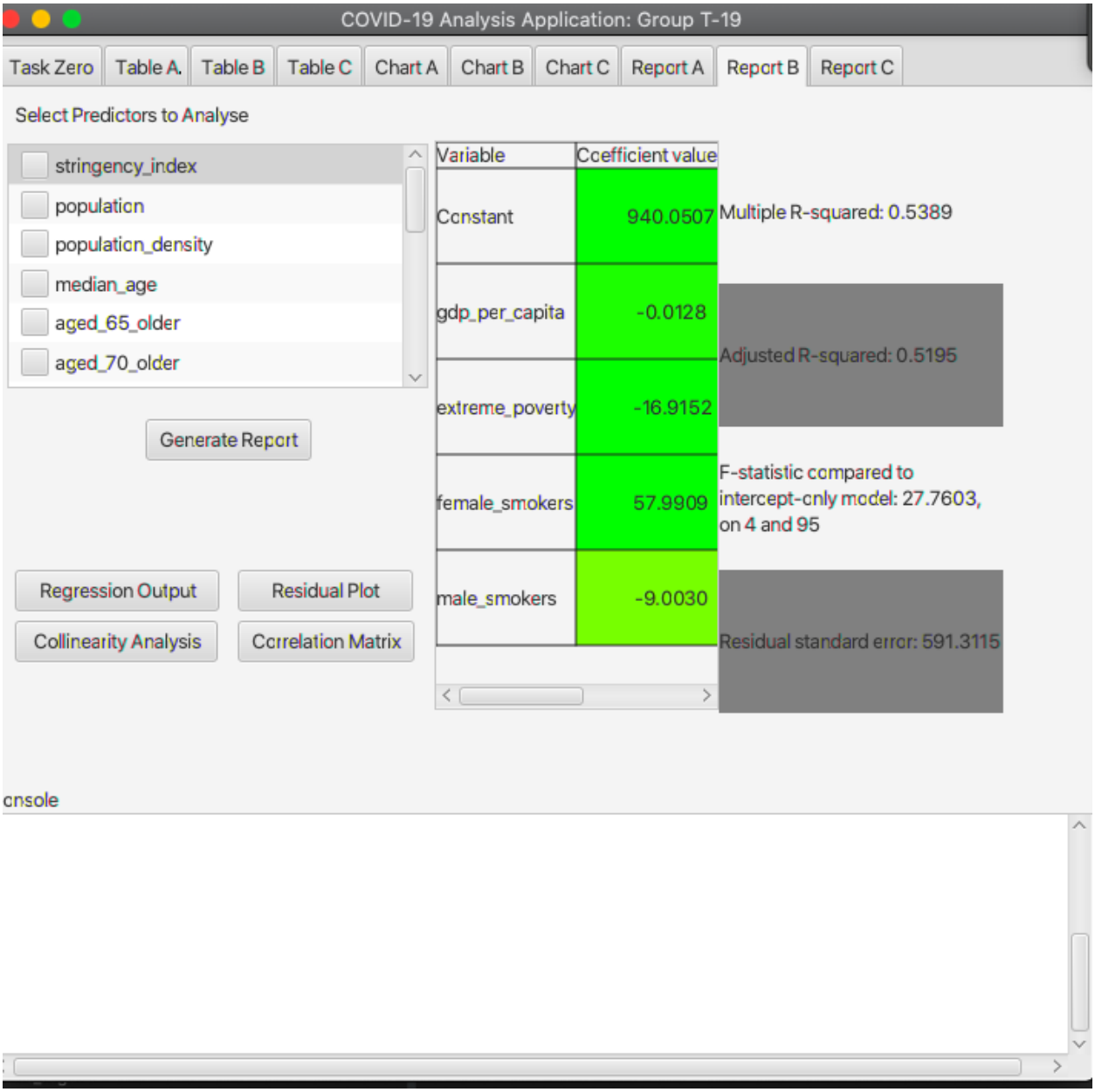


As we can see, the adjusted R-squared has risen slightly, which is good!

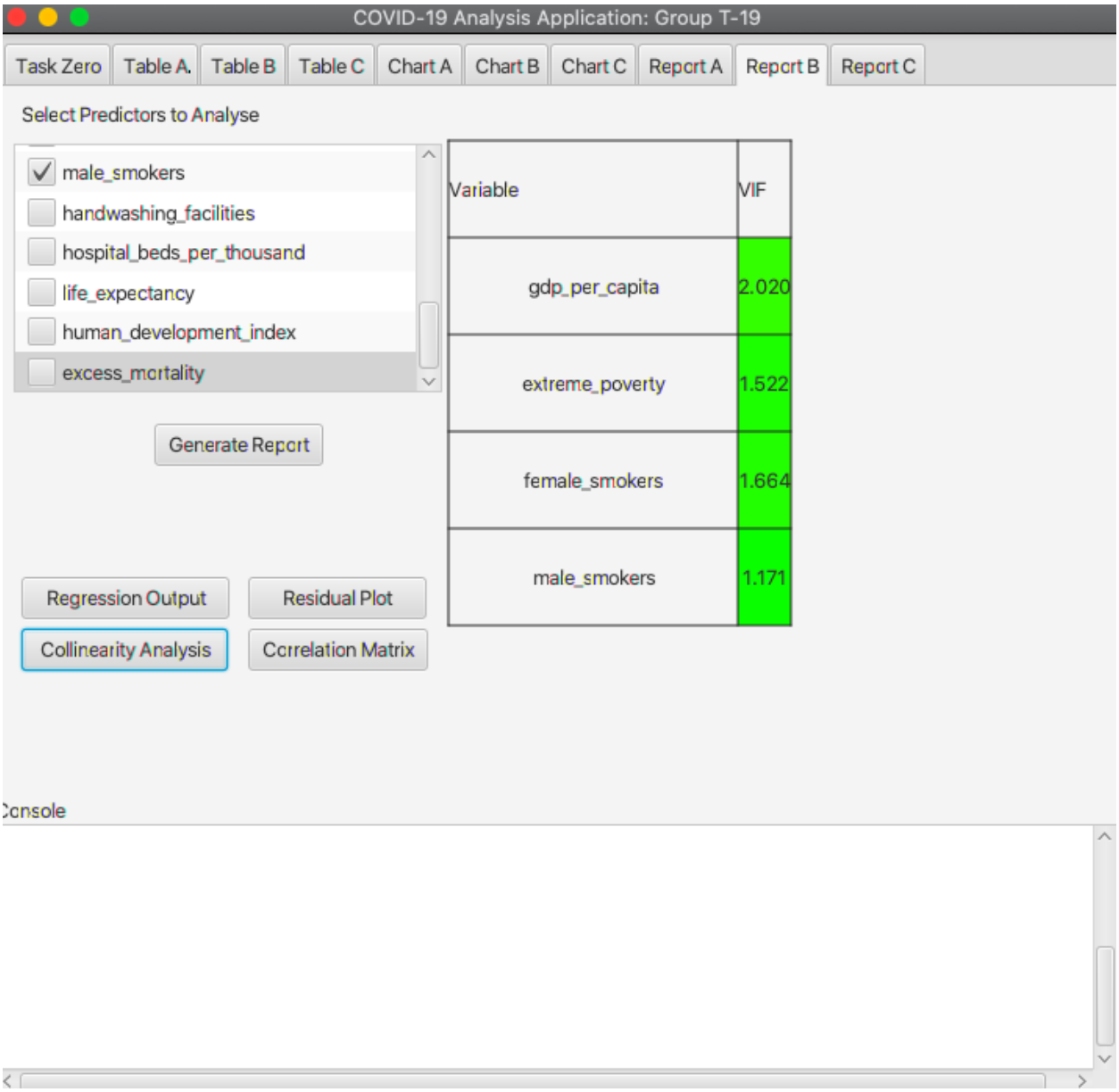


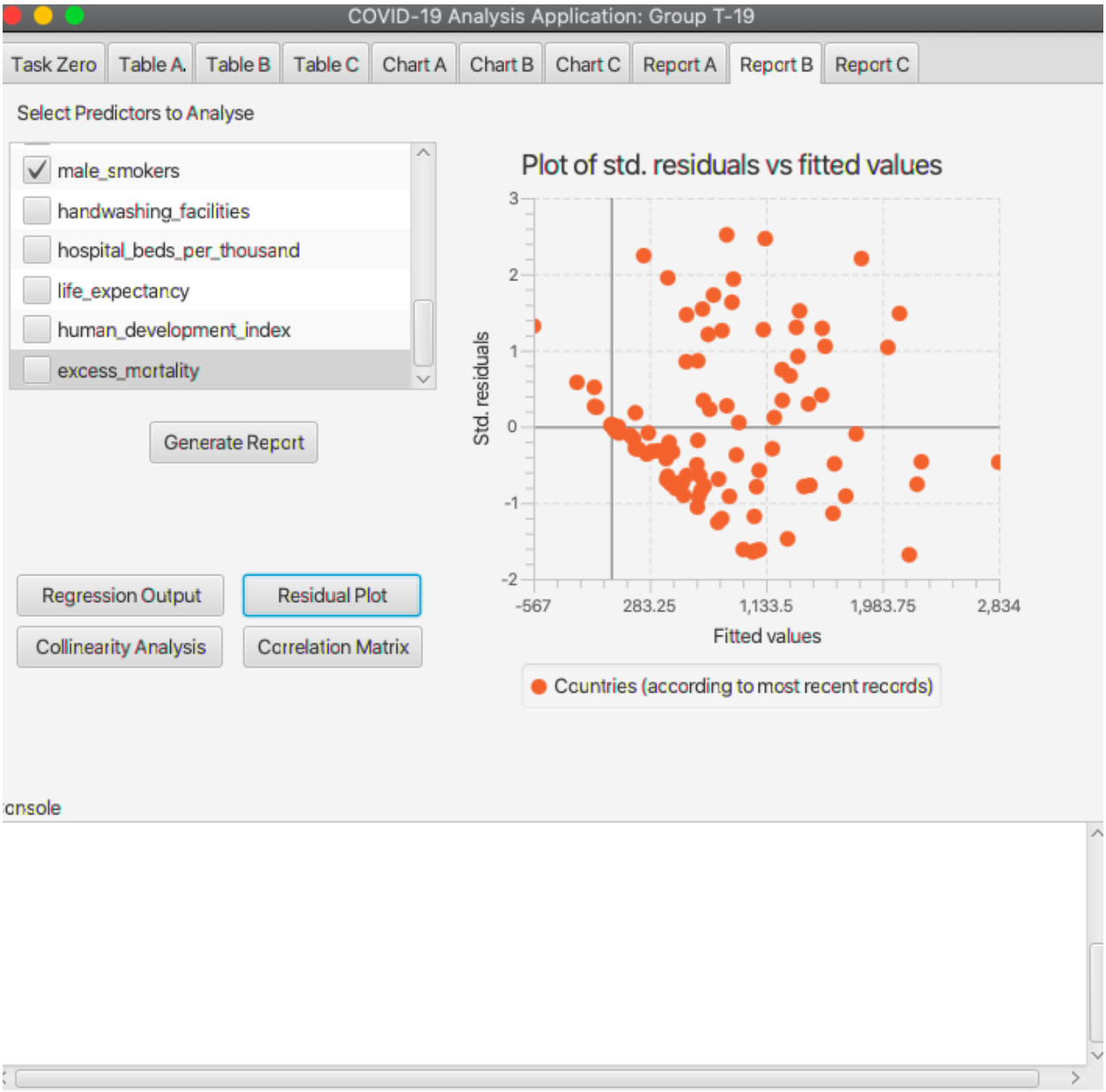
The VIFs are all now under 10, but the one for `life_expectancy` is a little troubling.

We may continue this process of dropping and adding variables to the model as we please. One of our best models (with the original, unmodified dataset) is:



This is because the residual plots show no significant pattern, all the VIFs are close to 1, and all regression coefficients are significant.


COVID-19 Analysis Application: Group T-19
Task Zero
Table A
Table B
Table C
Chart A
Chart B
Chart C
Report A
Report B
Report C
Select Predictors to Analyse
male_smokers
handwashing_facilities
hospital_beds_per_thousand
life_expectancy
human_development_index
excess_mortality
Generate Report
Regression Output
Residual Plot
Collinearity Analysis
Correlation Matrix
Variable
VIF
gdp_per_capita
2.020
extreme_poverty
1.522
female_smokers
1.664
male_smokers
1.171
Console



We could even add more predictor variables to the dataset and conduct analysis on them if we chose to!

Testing

Here is the R code used to write the tests in `RenjinGradleTest.java`.

```
setwd("/Users/jchow/Downloads/MATH3424 R") # need to set the starting directory as this

# RenjinGradle testing suite
test <- data.frame(X1=c(52.7, 109.3, 12.8, 66.8, 93.1, 115.4, -1),
                   X2=c(21.61, 41.51, 0.2, 26.2, 27.37, 43.22, 4.21),
                   X3=c(-1.05, -1.33, -1.49, -1.44, -3.22, -3.41, -22.1))

test[test==-1] <- NA
new_test <- as.data.frame(na.omit(test))
```

```r
print(new_test)
```

```
##      X1    X2    X3
## 1  52.7 21.61 -1.05
## 2 109.3 41.51 -1.33
## 3  12.8  0.20 -1.49
## 4  66.8 26.20 -1.44
## 5  93.1 27.37 -3.22
## 6 115.4 43.22 -3.41
```

```r
model = lm(X1 ~ ., data=new_test)
summary(model)
```

```
##
## Call:
## lm(formula = X1 ~ ., data = new_test)
##
## Residuals:
##      1      2      3      4      5      6
## -3.962  6.435  0.239 -2.981  7.260 -6.991
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.8446     7.8693   0.107  0.92130
## X2            2.2153     0.2339   9.473  0.00249 **
## X3           -7.5660     3.5162  -2.152  0.12051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.475 on 3 degrees of freedom
## Multiple R-squared:  0.9779, Adjusted R-squared:  0.9631
## F-statistic: 66.24 on 2 and 3 DF,  p-value: 0.003295
```

```r
cor(new_test)
```

```
##            X1         X2         X3
## X1  1.0000000  0.9714331 -0.5616557
## X2  0.9714331  1.0000000 -0.4041033
## X3 -0.5616557 -0.4041033  1.0000000
```

```r
library(car)
```

```
## Loading required package: carData
```

```r
vif(model)
```

```
##       X2       X3
## 1.195171 1.195171
```