STAT 331: Final Project

Ben Zhang, Frank Ding, Kevin Aby, Zafin Hassan

April 13, 2021

1 Summary

The main objective of this analysis was to create a linear regression model which best predicts mean leukocyte telomere length (LTL) in adults, using a dataset of observations drawn from $n = 864$ adults containing measurements of exposure to various persistent organic pollutants, biological factors such as the number of white blood cells, and social factors such as education level. We used the root mean squared prediction error (RMSE) as the metric for prediction accuracy.

We began by converting the categorical covariates into factors and summarizing the dataset with various numerical and graphical methods (eg boxplots, histograms, correlation matrix) and drew conclusions about the dataset, such as the presence of highly correlated covariates, which informed the rest of the analysis.

Then, we applied statistical methods to address multicollinearity in the data, transformed the data to satisfy the four linear regression assumptions, and performed automatic variable selection with cross-validation on stepwise methods, LASSO, and ridge regression to build three final models, which we compared against each other for the lowest RMSE. We found that the model fitted with cross-validation LASSO had the lowest RMSE of 0.2052.

In the context of the standard deviation of the dataset, this is a reasonably low error. However, we found that it was difficult to accurately predict mean LTL for observations with certain characteristics such as high organic pollutant concentrations. There were also several limitations in the analysis which could improve RMSE, if resolved.

2 Objective

The main objective of this analysis is to create a linear regression model which best predicts the outcome, mean leukocyte telomere length (LTL), using a dataset of observations drawn from $n = 864$ adults involving the following covariates: exposures to 18 different persistent organic pollutants (POP) (which includes 11 PCBs, 3 dioxins, and 4 furans), sex, age, education level, race, the number of years smoking cigarettes, whether the adult currently smokes, Body Mass Index (BMI), cotinine concentration, white blood cell count, and the percentage of these white blood cells which are lymphocytes, monocytes, eosinophils, basophils, and neutrophils.

Another objective is to analyze the dataset and identify relationships between covariates which may impact regression. We will resolve these issues by transforming the dataset, such as removing variables with high VIF.

A third objective is to apply statistical concepts, such as stepwise algorithms for automatic variable selection as well as shrinkage methods (LASSO and ridge regression), on a realistic dataset to assess their effectiveness in practice. As we iteratively improve our regression model, we perform model diagnosis to assess and correct any violations of the four assumptions of linear regression (linearity, independence, Normality, and homoscedasticity) and also consider possible outliers and influential points which are affecting the regression.

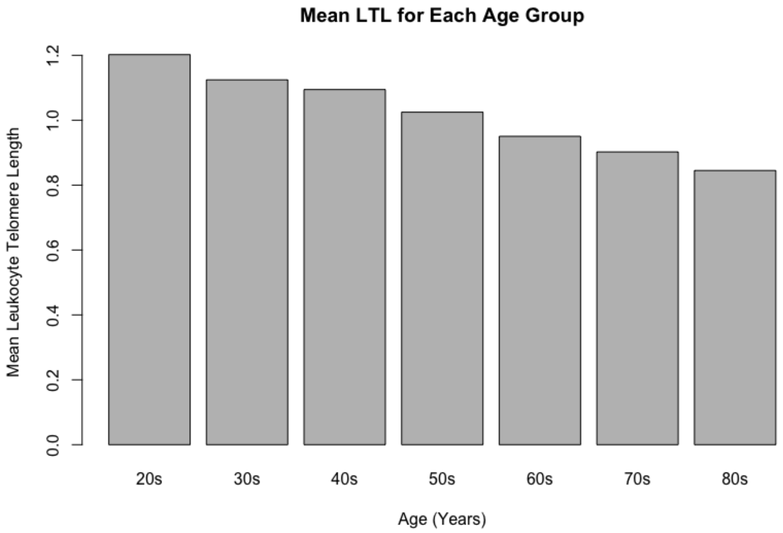
3 Exploratory Data Analysis

3.1 Summary Statistics

We begin by shuffling the data (with a seed of `12345` for reproducibility) and converting the four categorical covariates `male`, `edu_cat`, `race_cat`, `smokenow` into factors to prevent introducing assumptions about the magnitude of the numeric values used to represent the levels of these covariates in the data[1]. Then, we remove the indexing `X` column from the dataset, since it will not be used for EDA. The remaining 28 covariates, as well as the outcome, are all continuous and we do not make modifications to them for now. A five number summary for each of the continuous variables, and the number of observations at each level for each of the categorical variables can be found in Appendix 7.3.1. Since the LTL is very small (ranges from 0.5266 to 2.3512) relative to many covariates (eg the mean `POP_PCB4` observation is 38456), we expect any coefficient estimates for these covariates to also be very small.

3.2 Age

From the summary statistic for age, we see that the youngest adult in the dataset is 20 years old and the oldest is 85. We plot the mean LTL by age, grouped by decade[2],



It appears that the mean LTL is inversely related to age. We expect age to be a critical predictor in the rest of the analysis.

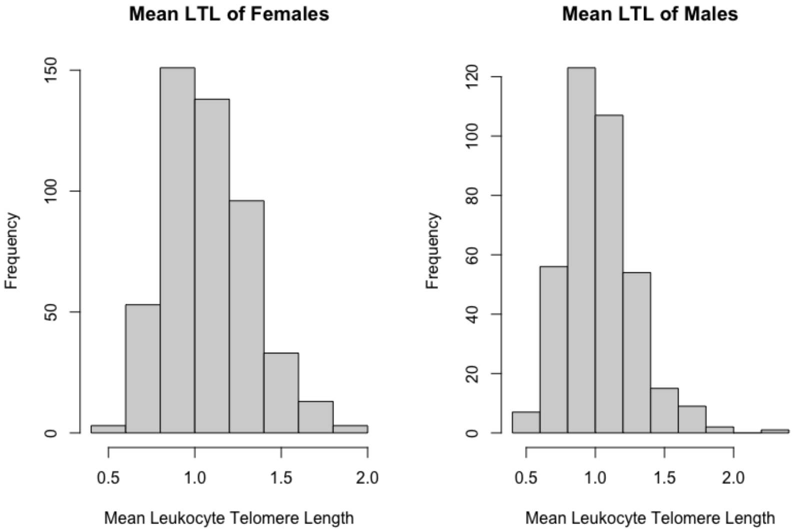
3.3 Sex

Due to the many known biological differences between males and females, we partition the dataset into male ($n_{male} = 374$) and female ($n_{female} = 490$) and plot the outcome by sex to determine whether a separate analysis for each sex is appropriate[3],

[1] See Appendix 7.1.1 for the R code

[2] See Appendix 7.1.2 for the R code

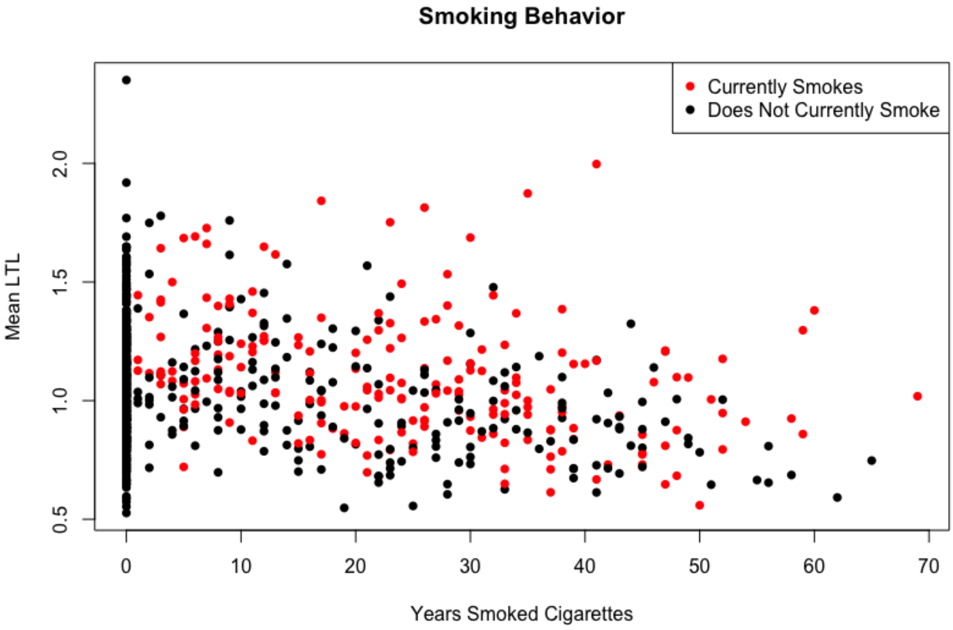
[3] See Appendix 7.1.3 for the R code



The distribution of LTL looks fairly similar between males and females. Furthermore, the mean and standard deviation of the outcomes for males is $1.0234, 0.2532$, respectively, and the mean and standard deviation of the outcomes for females is $1.0779, 0.2456$, respectively. Since these statistics are also quite similar, we will consider observations from both sexes simultaneously in the rest of the analysis.

3.4 Smoking

Two interesting covariates in the dataset are `yrssmoke` and `smokenow`, because unless there are people who have only been smoking for a few months, everyone who has zero years smoked will not be currently smoking. This relationship indicates some correlation between the two variables, which we further explore later. Plotting mean LTL against the number of years smoked and differentiating between those who are and are not currently smoking[4] verifies this,

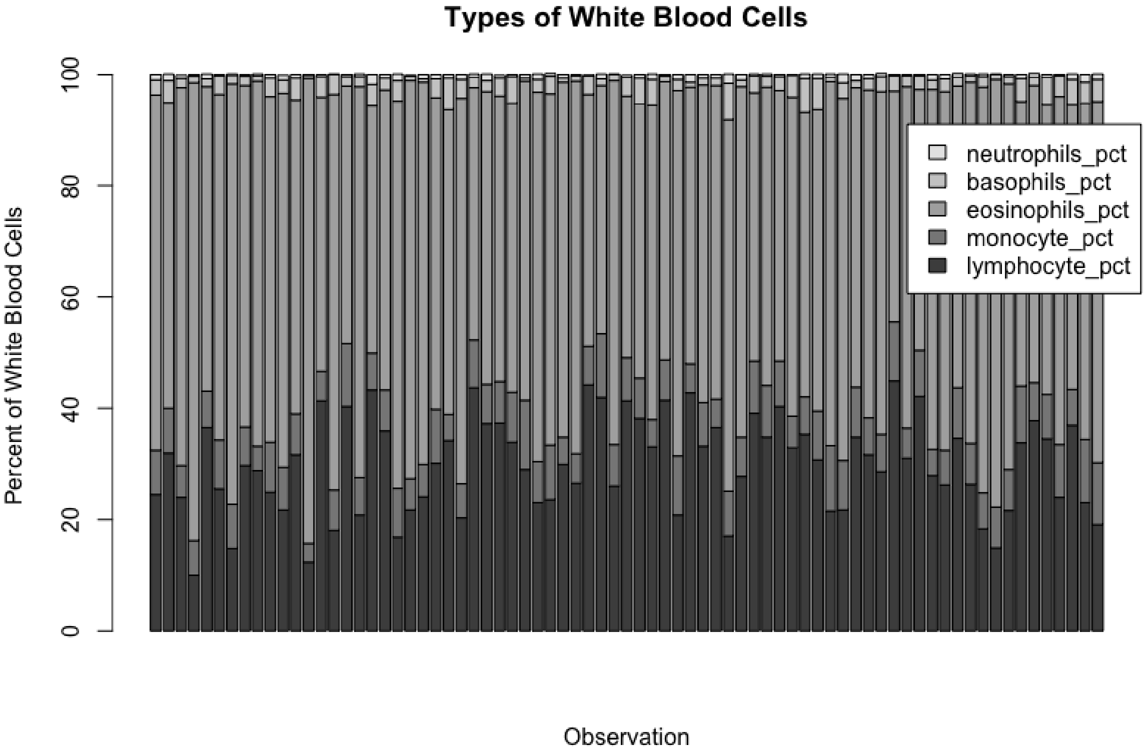


[4] See Appendix 7.1.4 for the R code

This plot also shows that there are many observations who no longer smoke but have smoked for decades in the past. Moreover, for these adults, there is no way to tell how many years ago they quit smoking. For these reasons, we expect `smokenow` to be a weak predictor and `yrssmoke` to be stronger. Another relationship to note, which hints at multicollinearity, is that for every observation, age is an upper bound on the number of years smoked.

3.5 White Blood Cells

We expect perfect, or at least very strong, multicollinearity among the five covariates for the percentages of lymphocytes, monocytes, eosinophils, basophils, and neutrophils in white blood cells, since these are all the types of white blood cells, so the five percentages should reasonably sum to 100 for each observation. Plotting the composition of the white blood cells for 75 observations[5] confirms this,



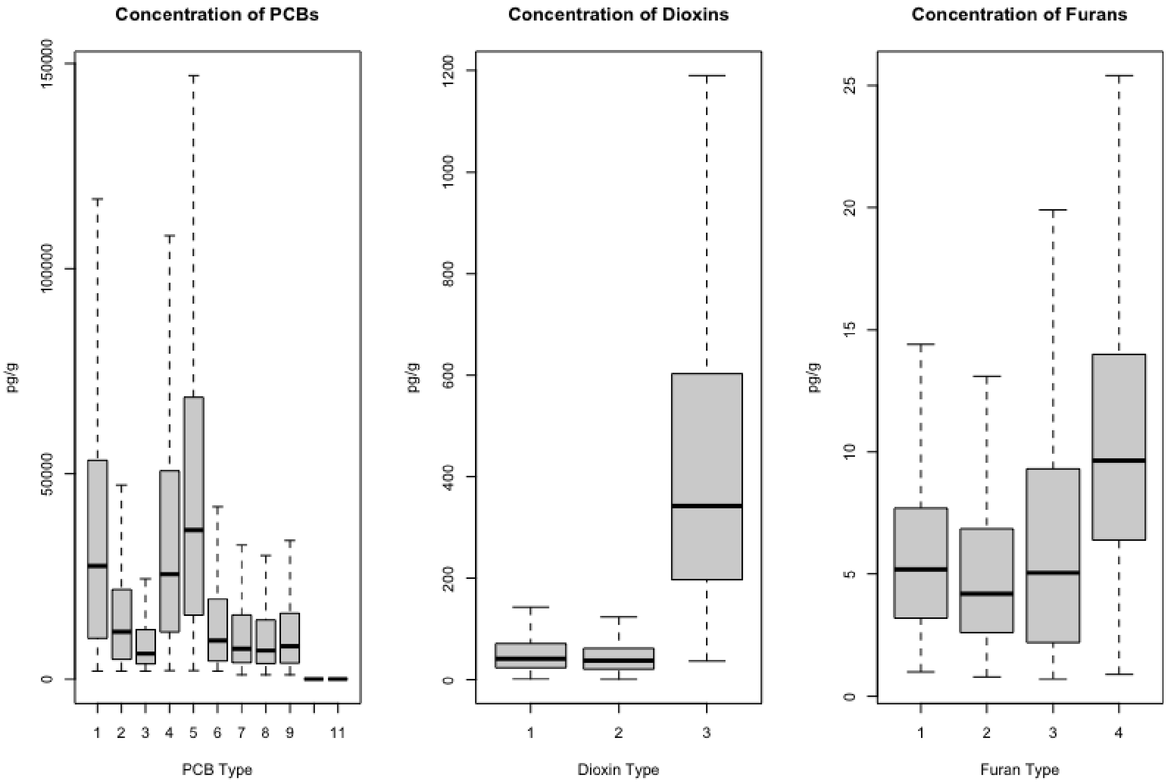
However, the five percentages do not sum to exactly 100% for every observation. For example, for observation #704, the percentages of lymphocyte, monocyte, eosinophils, basophils, neutrophils, respectively, are $23, 7.4, 66.4, 2.3, 1$, which sums to 100.1. This is likely due to rounding error in data entry. Fortunately, since there is no perfect multicollinearity, we will still be able to fit Least Squares models, but must deal with these very strongly correlated covariates during variable selection to prevent inflated variances of the coefficient estimators.

3.6 Exposures

Next, we consider the 11 PCBs, 3 dioxins, and 4 furans[6]. Plotting concentrations with box plot outliers removed,

[5] See Appendix 7.1.5 for the R code

[6] See Appendix 7.1.6 for the R code



At a glance, it appears the concentration in pg/g is much higher for `POP_dioxin3` than the other two in the observations, and there are similar patterns with certain PCBs and furans, although not as drastic. We will keep the varying concentrations of different types of organic pollutants in mind.

3.7 Correlation Among Continuous Covariates

Lastly, we investigate the correlation among the 28 continuous covariates and the mean LTL[7]. See Appendix 7.3.2 for the full correlation matrix plot. From this plot, we see that `eosinophils_pct` and `lymphocyte_pct` are very negatively correlated, which is in line with the fact that all five percentages of white blood cells sum to 100% (or very close to it) and from the earlier plot, `eosinophils_pct` and `lymphocyte_pct` are by far the two largest percentages of the five.

Secondly, it appears some of the organic pollutants are highly positively correlated with each other (for example, `POP_PCB1` and `POP_PCB2`). This indicates high multicollinearity among these organic pollutants, which we need to resolve later on.

Thirdly, age appears to be positively correlated with most of the organic pollutants. This makes intuitive sense since older adults likely have had more exposure to pollutants, either because they have simply been alive longer, or because of the lack of environmental and health protections from pollutants until modern times.

4 Methods

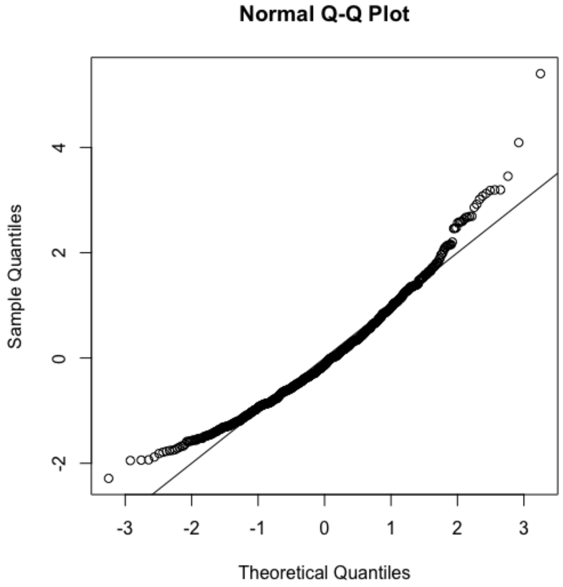
In this section, we transform the outcome to address issues with non-Normality, use an iterative algorithm to remove covariates with high VIF to address multicollinearity, fit three final models

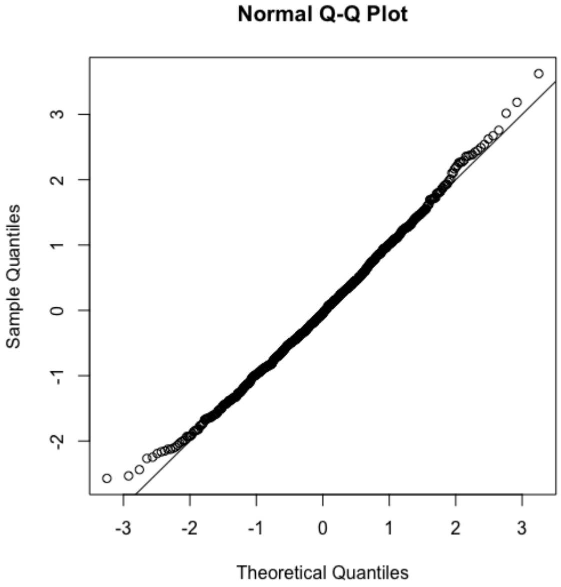
[7] See Appendix 7.1.7 for the R code

using stepwise selection, LASSO, and ridge regression, compare the RMSE of these three models, assess the four regression assumptions, and assess for outliers and influential points.

4.1 Transforming the Outcome

To begin, we fit a multiple regression model on the entire dataset, where the continuous covariates have been converted into factors[8]. However, we see that the Normality assumption does not hold on this full model, because the Q-Q plot of studentized residuals has two upwards-lifting tails[9] (left). This shape is similar to what one might expect with an Exponential distribution. We address this issue by taking the natural log of the outcome and refit[10]. The Q-Q plot of this new model (right) shows that the Normality assumption is satisfied.

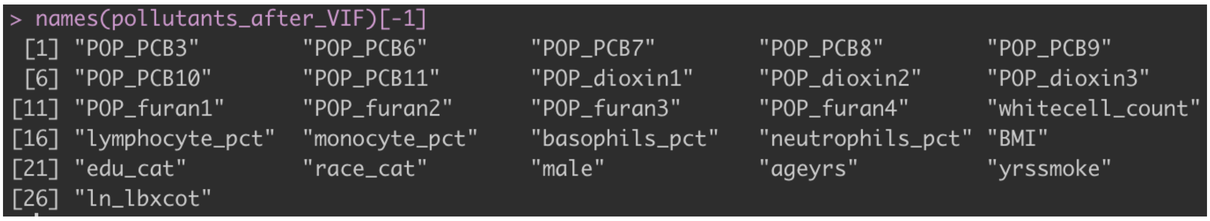




Henceforth, for model-building, we always take the natural log of the outcome. We continue to verify the other three regression assumptions later on.

4.2 Addressing Multicollinearity

As identified in the exploratory data analysis, there is very strong multicollinearity among the five covariates for white blood cell percentage as well as collinearity among other covariates such as `yrssmoke` and `smokenow`. To address this, we iteratively remove covariates with the highest VIF greater than 10 until all covariates have VIF less than 10, as in A3Q3[11]. The remaining 26 covariates are:



```
> names(pollutants_after_VIF)[-1]
 [1] "POP_PCB3"        "POP_PCB6"        "POP_PCB7"        "POP_PCB8"        "POP_PCB9"
 [6] "POP_PCB10"       "POP_PCB11"       "POP_dioxin1"     "POP_dioxin2"     "POP_dioxin3"
[11] "POP_furan1"      "POP_furan2"      "POP_furan3"      "POP_furan4"      "whitecell_count"
[16] "lymphocyte_pct"  "monocyte_pct"    "basophils_pct"   "neutrophils_pct" "BMI"
[21] "edu_cat"         "race_cat"        "male"            "ageyrs"          "yrssmoke"
[26] "ln_lbxcot"
```

[8] See Appendix 7.2.1 for the R code

[9] See Appendix 7.2.2 for the R code

[10] See Appendix 7.2.3 for the R code

[11] See Appendix 7.2.4 for the R code

Notably, `eosinophils_pct`, `smokenow`, and several PCB types have been removed, which resolves many of the multicollinearity issues discussed earlier.

Henceforth, for model-building, we use this subset of the dataset which addresses multicollinearity.

4.3 K-Fold Cross-Validation to Compare Stepwise Selection Algorithms

4.3.1 Procedure

We consider ten stepwise methods for automatic variable selection, of which five are forward and five are backward, and using five different λ values in the penalty term (where t is the number of parameters estimated),

$$AIC = -2 \log \mathcal{L}(\hat{\theta}) + \lambda t$$

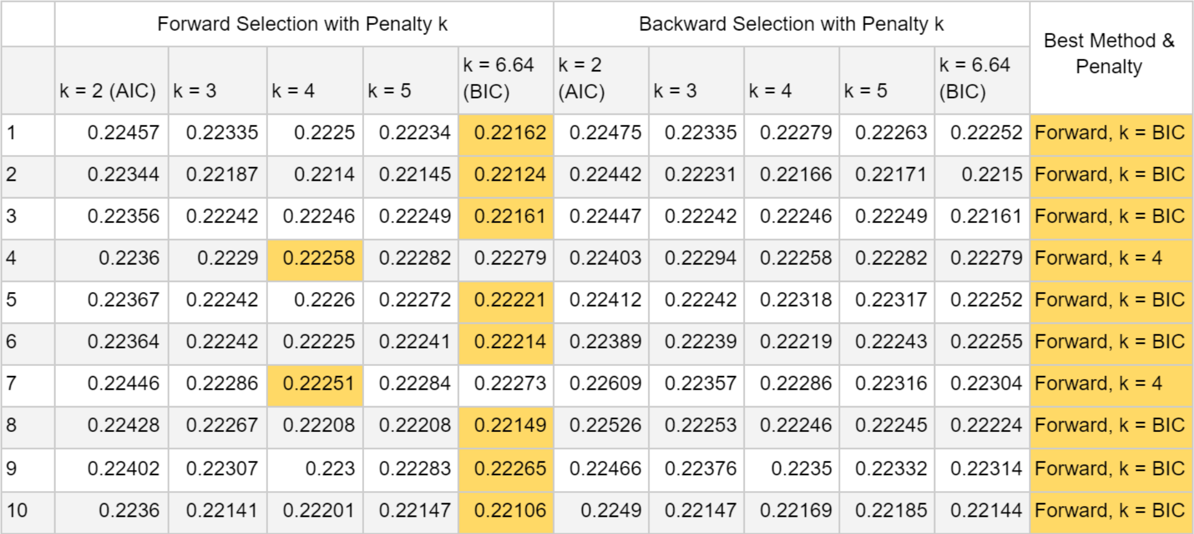
which are: $\lambda = 2$ (ie AIC), $3, 4, 5, \log(n)$ (ie BIC).

We use K-fold cross-validation (where $K = 9$) to choose the best method; that is, the method which produces the lowest mean $RMSE$. For each K-fold cross-validation process, the procedure is,

1. Partition the dataset into K folds
2. Use $K - 1$ folds as the training set and the kth as the validation set. Let $m = n/K = 864/9 = 96$ be the number of observations in each of the K fold
3. For each of the ten selection methods $i = 1, \ldots, 10$:
   (a) Build a model using this training set
   (b) Predict outcomes on the kth fold (denoted by the $m \times 1$ vector $\hat{\boldsymbol{y}}$) and compute,
   $$RMSE_{k,i} = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y_m - \hat{y}_m)^2}$$
4. Repeat steps 2 and 3, using each of the K folds as the validation set, and for each of the ten methods $i = 1, \ldots, 10$, take the mean of the K $RMSE$ as $RMSE_{CV,i}$,
   $$RMSE_{CV,i} = \frac{1}{K} \sum_{k=1}^{K} RMSE_k$$
5. Let,
   $$RMSE_{CV,best} = \min_i RMSE_{CV,i}$$
   and let i' be the selection method which achieves this minimum; call this the best method

Furthermore, we repeat this entire process ten times to derive stronger conclusions[12]. As seen in the following table, we find that although the errors are similar across all ten methods, forward stepwise selection with the BIC penalty produces the smallest root mean squared error in eight of ten repetitions,

[12] See Appendix 7.2.5 for the R code



	Forward Selection with Penalty k					Backward Selection with Penalty k					Best Method & Penalty
	k = 2 (AIC)	k = 3	k = 4	k = 5	k = 6.64 (BIC)	k = 2 (AIC)	k = 3	k = 4	k = 5	k = 6.64 (BIC)	
1	0.22457	0.22335	0.2225	0.22234	0.22162	0.22475	0.22335	0.22279	0.22263	0.22252	Forward, k = BIC
2	0.22344	0.22187	0.2214	0.22145	0.22124	0.22442	0.22231	0.22166	0.22171	0.2215	Forward, k = BIC
3	0.22356	0.22242	0.22246	0.22249	0.22161	0.22447	0.22242	0.22246	0.22249	0.22161	Forward, k = BIC
4	0.2236	0.2229	0.22258	0.22282	0.22279	0.22403	0.22294	0.22258	0.22282	0.22279	Forward, k = 4
5	0.22367	0.22242	0.2226	0.22272	0.22221	0.22412	0.22242	0.22318	0.22317	0.22252	Forward, k = BIC
6	0.22364	0.22242	0.22225	0.22241	0.22214	0.22389	0.22239	0.22219	0.22243	0.22255	Forward, k = BIC
7	0.22446	0.22286	0.22251	0.22284	0.22273	0.22609	0.22357	0.22286	0.22316	0.22304	Forward, k = 4
8	0.22428	0.22267	0.22208	0.22208	0.22149	0.22526	0.22253	0.22246	0.22245	0.22224	Forward, k = BIC
9	0.22402	0.22307	0.223	0.22283	0.22265	0.22466	0.22376	0.2235	0.22332	0.22314	Forward, k = BIC
10	0.2236	0.22141	0.22201	0.22147	0.22106	0.2249	0.22147	0.22169	0.22185	0.22144	Forward, k = BIC

Thus, we consider forward selection with the BIC penalty the best stepwise selection method.

We then use it to build a model on the entire dataset[13]; after forward selection, only `ageyrs` and `POP_furan3` are left. Denote this model $M_{stepbest}$,

$$M_{stepbest} : \texttt{LTL} = \beta_0 + \beta_1\texttt{ageyrs} + \beta_2\texttt{POP\_furan3} + \epsilon$$

4.3.2 Stepwise Model Diagnostics

We now investigate whether the four regression assumptions hold for $M_{stepbest}$,

1. Linearity: To check the linearity assumption, we look at added variable plots[14]. In both plots (see Appendix 7.3.3), there is no discernible linear pattern. Thus, the linearity assumption is satisfied.

2. Normality: Although we already corrected a violation of the Normality assumption earlier by taking the log of the outcome, we check this assumption again on $M_{stepbest}$[15] and verify that the Q-Q plot of the studentized residuals still shows Normality (see Appendix 7.3.4).

3. Homoskedasticity: To check the homoskedasticity assumption, we plot studentized residuals against fitted values[16] (see Appendix 7.3.5). There does not appear to be a mean-variance relationship, so the homoskedasticity assumption holds.

4. Independence: Without more information on how the data was collected, we assume there was no clustered sampling and that the independence assumption holds. This is a limitation of the analysis further explained in the Discussion section.

Since there are no violations of any of the four assumptions, $M_{stepbest}$ does not need to be further modified.

4.3.3 Stepwise Model Fit

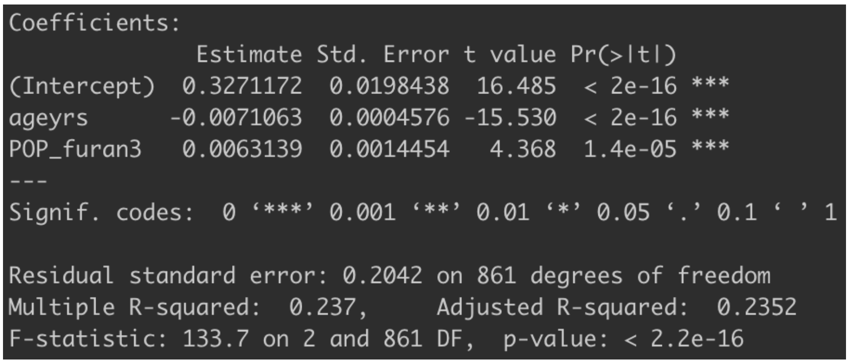
Overall, the estimated coefficients, $R^2, R^2_{adj}, \hat{\sigma}$ in $M_{stepbest}$ are,

[13] See Appendix 7.2.6 for the R code

[14] See Appendix 7.2.7 for the R code

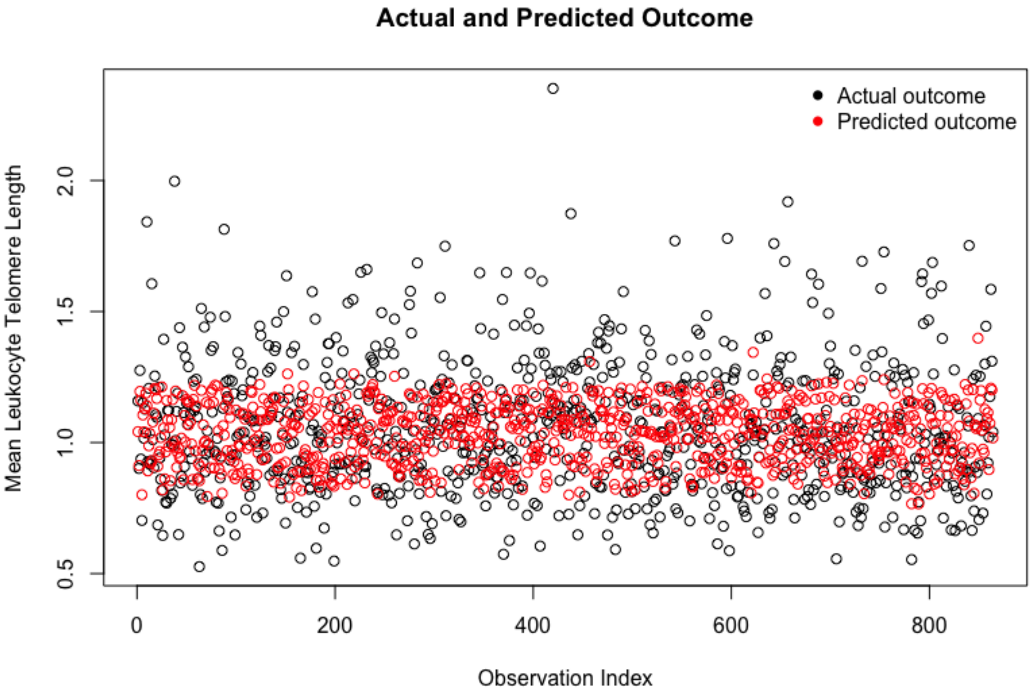
[15] See Appendix 7.2.8 for the R code

[16] See Appendix 7.2.9 for the R code



```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.3271172  0.0198438  16.485  < 2e-16 ***
ageyrs      -0.0071063  0.0004576 -15.530  < 2e-16 ***
POP_furan3   0.0063139  0.0014454   4.368  1.4e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2042 on 861 degrees of freedom
Multiple R-squared:  0.237,	Adjusted R-squared:  0.2352
F-statistic: 133.7 on 2 and 861 DF,  p-value: < 2.2e-16
```

Obtaining predictions with $M_{stepbest}$ and plotting them against the actual outcomes[17],



The RMSE is 0.2231, which is comparable to 0.2502, the standard deviation of the 864 observed outcomes. However, from the plot, it appears large values in mean LTL are especially poorly predicted by the model. We investigate whether any of these points are outliers and/or influential later on.

4.4 LASSO and Ridge Regression with Cross-Validation

Next, we use LASSO and ridge regression methods with cross-validation for automatic variable selection[18]. We partitioned the 864 total observations into 600 for the training set (for fitting the two models) and 264 for the test set (for evaluating prediction accuracy) then used `cv.glmnet`. The cross-validation is to select a λ which minimizes $RMSE$.

We obtained a $RMSE$ of 0.2052 for the final LASSO model (denoted M_{LASSO}) and 0.2064 for the final ridge regression model (denoted M_{Ridge}). See Appendix 7.3.6 to see the coefficient estimates in M_{LASSO} (left) and M_{Ridge} (right). As expected, since the mean LTL is relatively small compared to the covariates, the non-zero coefficients in both models are also all very small.

[17] See Appendix 7.2.10 for the R code

[18] See Appendix 7.2.11 for the R code

Plots for the cross-validation process to select a λ which minimizes the $RMSE$ can be found in Appendix 7.3.7 and 7.3.8.

We analyze the prediction accuracy of these models and compare them with $M_{stepbest}$ further in the Results section. Due to limitations in time, we were unable to perform diagnostics on the `glmnet` objects. However, we expect they follow the four assumptions reasonably, as $M_{stepbest}$ did.

4.5 Outliers and Influence

Having built three models $M_{stepbest}, M_{LASSO}, M_{Ridge}$, we now investigate whether there are x-outliers, y-outliers, or influential points in the dataset. For simplicity, we use $M_{stepbest}$ to analyze this, although results should generalize to the other two models. Recall $M_{stepbest}$ regresses the log of the mean LTL on `ageyrs` and `POP_furan3`.

First, we plot leverage and consider points x-outliers (high leverage) if its leverage is greater than twice the mean leverage[19] (see Appendix 7.3.9 for the plot; high leverage points are colored red). In total, there are 55 high leverage points.

Next, we check whether any of these are y-outliers by plotting absolute studentized jackknife residuals[20] (see Appendix 7.3.10 for the plot; the high leverage points are colored red). From this plot, we see that although there are a few points which are both high leverage and have high absolute studentized jackknife residuals, none stand out as especially high and the few which are higher are not high leverage points.

More importantly, we consider whether any points are influential using DFFITS, Cook's distance, and DFBETAS[21]. See Appendix 7.3.11 for plots of the three criteria respectively. From these plots, we see there are observations such as #849 which are considered influential points according to all three criteria.

Using the influential points identified by each of the three criteria, we plot the predicted outcomes against `ageyrs` (which is a strong predictor, as discussed earlier) with and without these points[22] (see Appendix 7.3.12 for the three plots). We see that even though there are points which are considered influential by one or more of the above criteria, in context, the predictions look similar for models fitted with and without these points for all three criteria.

Overall, even though there are several high-leverage points, they do not appear to be y-outliers, and even though there were influential points, they do not significantly affect our regression models.

5 Results

The RMSE of the three final models $M_{stepbest}, M_{LASSO}, M_{Ridge}$ are $0.2231, 0.2052, 0.2064$, respectively. In comparison, the standard deviation of the observed outcomes in the entire dataset is 0.2502 and of the outcomes in the test set is 0.2289. Both M_{LASSO} and M_{Ridge} achieve lower RMSE than these sample standard deviations.

As well, even though the models fitted with LASSO and ridge regression achieve a lower RMSE, the model obtained by forward selection is also reasonably accurate in context and may be preferred because it is simpler and regresses on much fewer (only two) covariates, compared to 21 in the LASSO model and 24 in the ridge regression model.

[19] See Appendix 7.2.12 for the R code

[20] See Appendix 7.2.13 for the R code

[21] See Appendix 7.2.14 for the R code

[22] See Appendix 7.2.14 for the R code

6 Discussion

6.1 Conclusions

In conclusion, we found that our three final models can be ordered, from lowest (best) RMSE to highest, as: $M_{LASSO}, M_{Ridge}, M_{stepbest}$. The RMSE of all three models was lower than the standard deviation of the observed outcomes; they are able to predict the outcome reasonably well. As well, the statistical methods learned in STAT 331, such as the iterative algorithm to remove covariates with high VIF and taking the log of the outcome to resolve non-Normality, were effective at dealing with these issues in a realistic dataset.

One surprising point is that only one POP covariate remained after forward selection in $M_{stepbest}$, even after adjusting for multicollinearity by removing covariates with high VIF. This indicates that all the POP covariates except `POP_furan3` are weak predictors. This is also the case in both M_{LASSO} and M_{Ridge}, where the coefficients of several POP covariates shrunk to zero.

A covariate present in all three models was `ageyrs`. This aligns with our findings in the exploratory data analysis that age could be a strong predictor for mean LTL. This makes biological sense, since LTL is longest at birth and decreases progressively with age, due to cell reproduction.

A last point to note is that due to the absolute value of the mean LTL being very small relative to most covariate values, the coefficient estimates for any covariates (ie all excluding $\hat{\beta}_0$) in all three models are quite small.

6.2 Limitations

There were several limitations with our analysis. Due to limitations in time, we were unable to assess for y-outliers and influential points using our best model (M_{LASSO}) and instead, used $M_{stepbest}$ as a proxy. We expect similar results anyway, as seen in how relatively close the $RMSE$ are between the two models. Secondly, we used RMSE as the sole metric for prediction accuracy. An extension could be to create prediction intervals and look at the proportion of intervals which actually contained the observed outcome.

As well, we did not look at interaction terms or other clever ways to transform the data, besides removing covariates to deal with multicollinearity and taking the log of the outcome to deal with non-Normality. This would have perhaps improved prediction accuracy by being able to represent more complex relationships between the covariates. Furthermore, we did not do quantitative hypothesis tests (eg t-tests or F-tests) to test for significance, which would have been useful to strengthen our conclusions. Lastly, due to the lack of information about how the data was collected and the lack of time-series data, we were unable to assess the independence assumption. Our analysis may be impacted if this assumption is in fact violated.

7 Appendix

7.1 R Code for Exploratory Data Analysis

7.1.1 Parsing Data

```
get_data <- function(remove_indices = TRUE) {
  pollutants <- read.csv("pollutants.csv")
  if (shuffle) {
    # Set a seed so that the shuffle is reproducible
    set.seed(12345)
    pollutants <- pollutants[sample(nrow(pollutants)),]
  }
  if (remove_indices) {
    # Remove index column
    pollutants[,1] <- NULL
  }
  # Process categorical covariates
  male_levels <- c("Female", "Male")
  pollutants$male <- factor(pollutants$male, levels=c(0,1),
                            labels = male_levels)
  edu_cat_levels <- c("NoHS", "HS/GED", "College/AA", "CollegeGrad")
  pollutants$edu_cat <- factor(pollutants$edu_cat, levels=c(1,2,3,4),
                               labels = edu_cat_levels)
  race_cat_levels <- c("Other", "Mexican", "Black", "White")
  pollutants$race_cat <- factor(pollutants$race_cat, levels=c(1,2,3,4),
                                labels = race_cat_levels)
  smokenow_levels <- c("No", "Yes")
  pollutants$smokenow <- factor(pollutants$smokenow, levels=c(0,1),
                                labels = smokenow_levels)
  return(pollutants)
}

# Get the dataset
pollutants <- get_data()
```

7.1.2 LTL by Age

```
mean_length_age = c()
for (i in 2:8){
  temp = filter(pollutants, ageyrs %in% ((i*10):(((i+1)*10)-1)))
  mean_length_age[i-1] = mean(temp$length)
}
names(mean_length_age) = c(’20s’,’30s’,’40s’,’50s’,’60s’,’70s’,’80s’)
barplot(mean_length_age, xlab = ’Age (Years)’,
        ylab = ’Mean Leukocyte Telomere Length’,
        main = ’Mean LTL for Each Age Group’)
```

7.1.3 LTL by Sex

```
female_data <- filter(pollutants, male == "Female")
male_data <-  filter(pollutants, male == "Male")
par(mfrow = c(1,2))
# The distribution of length is similar between males and females
hist(female_data[,1], ylab = "Frequency", xlab = ’Mean Leukocyte Telomere Length’,
     main = "Mean LTL of Females")
hist(male_data[,1], ylab = "Frequency", xlab = ’Mean Leukocyte Telomere Length’,
     main = "Mean LTL of Males")
sex_mean_sd = c(mean(male_data$length), mean(female_data$length),
                sd(male_data$length), sd(female_data$length))
```

7.1.4 LTL by Smoking Status

```
par(mfrow=c(1,1))
plot(pollutants$length~pollutants$yrssmoke, ylab="Mean LTL",
     xlab="Years Smoked Cigarettes", main = "Smoking Behavior", pch=16,
     col=ifelse(pollutants$smokenow == "Yes", "red", "black"))
legend(x = "topright",
       legend = c("Currently Smokes", "Does Not Currently Smoke"),
       col = c("red", "black"), pch = c(16, 16))
```

7.1.5 White Blood Cells Breakdown

```
covariates <- names(pollutants)
pcts <- covariates[grepl("_pct", covariates)]
white_blood_cells_pcts <- as.matrix(pollutants[1:75, pcts])
barplot(t(white_blood_cells_pcts), xaxt="n", legend.text=TRUE,
        main="Types of White Blood Cells",
        xlab="Observation", ylab="Percent of White Blood Cells",
        args.legend=list(x="topright", inset = c(0, 0.09)))
```

7.1.6 Exposures

```
PCBs <- covariates[grepl("POP_PCB", covariates)]
dioxins <- covariates[grepl("POP_dioxin", covariates)]
furans <- covariates[grepl("POP_furan", covariates)]
par(mfrow=c(1,3))
boxplot(pollutants[, names(pollutants) %in% PCBs], outline = FALSE,
        xlab = "PCB Type", ylab = "pg/g",
        main = "Concentration of PCBs", xaxt = "n")
axis(1, at = 1:11, labels = c(1:11))
boxplot(pollutants[, names(pollutants) %in% dioxins], outline = FALSE,
        xlab = "Dioxin Type", ylab = "pg/g",
        main = "Concentration of Dioxins", xaxt = "n")
axis(1, at = 1:3, labels = c(1:3))
boxplot(pollutants[, names(pollutants) %in% furans], outline = FALSE,
        xlab = "Furan Type", ylab = "pg/g",
        main = "Concentration of Furans", xaxt = "n")
axis(1, at = 1:4, labels = c(1:4))
```

7.1.7 Correlation Among Covariates

```
library(ggcorrplot)
categoricals <- c("male", "edu_cat", "race_cat", "smokenow")
continuous_data <- pollutants[, !names(pollutants) %in% categoricals]
corr <- round(cor(continuous_data), 1)
ggcorrplot(corr, tl.cex = 6)
```

7.2 R Code for Model Building

7.2.1 Fitting the Full Model

```
get_data <- function(remove_indices = TRUE, shuffle = TRUE) {
  pollutants <- read.csv("pollutants.csv")
  if (shuffle) {
    # Set a seed so that the shuffle is reproducible
    set.seed(12345)
    pollutants <- pollutants[sample(nrow(pollutants)),]
  }
  if (remove_indices) {
    # Remove index column
    pollutants[,1] <- NULL
  }
  # Process categorical covariates
  male_levels <- c("Female", "Male")
  pollutants$male <- factor(pollutants$male, levels=c(0,1),
                            labels = male_levels)
  edu_cat_levels <- c("NoHS", "HS/GED", "College/AA", "CollegeGrad")
  pollutants$edu_cat <- factor(pollutants$edu_cat, levels=c(1,2,3,4),
                               labels = edu_cat_levels)
  race_cat_levels <- c("Other", "Mexican", "Black", "White")
  pollutants$race_cat <- factor(pollutants$race_cat, levels=c(1,2,3,4),
                                labels = race_cat_levels)
  smokenow_levels <- c("No", "Yes")
  pollutants$smokenow <- factor(pollutants$smokenow, levels=c(0,1),
                                labels = smokenow_levels)
  return(pollutants)
}

pollutants <- get_data()
Mfull <- lm(length~., data=pollutants)
```

7.2.2 Check Normality on the Full Model

```
check_normality <- function(Model) {
  res1 <- resid(Model)
  # studentized residuals
  stud1 <- res1/(sigma(Model)*sqrt(1-hatvalues(Model)))
  # qqplot of studentized residuals
  qqnorm(stud1)
  # add 45 degree line
  abline(0,1)
}

check_normality(Mfull)
```

7.2.3 Log Outcome and Check Normality Again

```
Mfull_log <- lm(log(length)~., data=pollutants)
check_normality(Mfull_log)
```

7.2.4 Remove High VIF Covariates

```
library(car)
# Algorithm from A3Q3: iteratively remove covariates with high VIF > 10
eliminate_by_VIF <- function(dataset) {
  Mfull <- lm(log(length)~., data = dataset)
  M_vif <- Mfull
  for (i in 1:length(Mfull$coefficients)) {
    VIF <- vif(M_vif)
    if (max(VIF) > 10) {
      M_vif <- update(M_vif, paste('~. -', names(coef(M_vif))[which.max(VIF) + 1]))
    } else {
      break
    }
  }
  return(M_vif)
}
Mfull_after_VIF <- eliminate_by_VIF(pollutants)
names(coef(Mfull_after_VIF)
# Update the data set itself by removing columns correpsonding to covariates
# which have been removed because of high VIF. Categorical covariates
# need to be explicitly added back in because only certain levels remain.
pollutants_after_VIF <- pollutants[, names(pollutants) %in%
                                     c("length", "male", "edu_cat", "race_cat",
                                       names(coef(Mfull_after_VIF)))]
```

7.2.5 K-Fold Cross Validation to Compare Stepwise Selection Algorithms

```
library(MASS)

K <- 9 # Number of folds
N <- nrow(pollutants_after_VIF) # Number of observations

# vector to hold all RMSEcv values for different models,
# repeating K-fold cross validation 10 times.
RMSE_data <- c()

# Loop repeating K-fold cross validation 10 times.
```

```
for(i in 1:10) {

  # Sampling 1:N into K folds (i.e. approx. 864/K 1’s, 864/K 2’s, ..., 864/K K’s)
  validSetSplits <- sample((1:N)%%K + 1)

  # Each vector contains K RMSE values, i.e. one for each of the K folds serving
  # as the validation set.
  # Example: RMSE1 contains 9 RMSEcv values for the forward stepwise method and
  # AIC penalty, one for each of the 9 folds serving as the validation set

  RMSE1 <- c() # Forward stepwise with AIC penalty
  RMSE2 <- c() # Forward stepwise with penalty = 3
  RMSE3 <- c() # Forward stepwise with penalty = 4
  RMSE4 <- c() # Forward stepwise with penalty = 5
  RMSE5 <- c() # Forward stepwise with BIC penalty

  RMSE6 <- c() # Backward stepwise with AIC penalty
  RMSE7 <- c() # Backward stepwise with penalty = 3
  RMSE8 <- c() # Backward stepwise with penalty = 4
  RMSE9 <- c() # Backward stepwise with penalty = 5
  RMSE10 <- c() # Backward stepwise with BIC penalty

  for(k in 1:K) { # doing K fold cross validation

    # Setting Validation and Test Dataset
    validSet <- pollutants_after_VIF[validSetSplits == k,]
    trainSet <- pollutants_after_VIF[validSetSplits != k,]

    full <- lm(length ~ ., data = trainSet)
    empty <- lm(length ~ 1, data = trainSet)

    m1 <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
                  direction = "forward")
    pred1 <- predict(m1, newdata = validSet)
    RMSE1[k] <- sqrt(mean((validSet$length - pred1)^2))

    m2 <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
                  direction = "forward", k = 3)
    pred2 <- predict(m2, newdata = validSet)
    RMSE2[k] <- sqrt(mean((validSet$length - pred2)^2))

    m3 <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
                  direction = "forward", k = 4)
    pred3 <- predict(m3, newdata = validSet)
    RMSE3[k] <- sqrt(mean((validSet$length - pred3)^2))

    m4 <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
                  direction = "forward", k = 5)
    pred4 <- predict(m4, newdata = validSet)
    RMSE4[k] <- sqrt(mean((validSet$length - pred4)^2))
    m5 <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
                  direction = "forward", k = log(nrow(trainSet)))
    pred5 <- predict(m5, newdata = validSet)
    RMSE5[k] <- sqrt(mean((validSet$length - pred5)^2))

    m6 <- stepAIC(object =  full, scope = list(upper = full, lower = empty),
                  direction = "backward")
    pred6 <- predict(m6, newdata = validSet)
    RMSE6[k] <- sqrt(mean((validSet$length - pred6)^2))

    m7 <- stepAIC(object =  full, scope = list(upper = full, lower = empty),
                  direction = "backward", k = 3)
    pred7 <- predict(m7, newdata = validSet)
    RMSE7[k] <- sqrt(mean((validSet$length - pred7)^2))

    m8 <- stepAIC(object =  full, scope = list(upper = full, lower = empty),
                  direction = "backward", k = 4)
    pred8 <- predict(m8, newdata = validSet)
    RMSE8[k] <- sqrt(mean((validSet$length - pred8)^2))

    m9 <- stepAIC(object =  full, scope = list(upper = full, lower = empty),
                  direction = "backward", k = 5)
    pred9 <- predict(m9, newdata = validSet)
    RMSE9[k] <- sqrt(mean((validSet$length - pred9)^2))

    m10 <- stepAIC(object =  full, scope = list(upper = full, lower = empty),
                   direction = "backward", k = log(nrow(trainSet)))
    pred10 <- predict(m10, newdata = validSet)
    RMSE10[k] <- sqrt(mean((validSet$length - pred10)^2))
  }

  # Storing all the RMSEcv values in a vector to compare and find best selection
  # method and penalty
  RMSE_data <- c(RMSE_data, mean(RMSE1), mean(RMSE2), mean(RMSE3), mean(RMSE4),
                 mean(RMSE5), mean(RMSE6), mean(RMSE7), mean(RMSE8), mean(RMSE9),
                 mean(RMSE10))

}
```

7.2.6 Best Stepwise Model

```
start <- lm(length ~ 1, data = pollutants_after_VIF)
end <- lm(length ~ ., data = pollutants_after_VIF)
M_stepbest <- stepAIC(object = start, scope = list(upper = end, lower = start),
                      direction = "forward", k = log(N))
M_stepbest <- lm(log(length) ~ ageyrs + POP_furan3, data = pollutants_after_VIF)
summary(M_stepbest)
```

7.2.7 Check Linearity for Stepwise Model

```
check_linearity <- function(Model) {
  avPlots(Model)
}

check_linearity(M_stepbest)
```

7.2.8 Check Normality for Stepwise Model

```
check_normality(M_stepbest)
```

7.2.9 Check Homoskedasticity for Stepwise Model

```
check_heteroskedasticity <- function(Model) {
  res1 <- resid(Model) # raw residuals
  stud1 <- res1/(sigma(Model)*sqrt(1-hatvalues(Model))) # studentized residuals
  ## plot of studentized residuals vs fitted values
  plot(stud1~fitted(Model),
       xlab="Fitted Vals",
       ylab="Studentized Residuals",
       main="Residuals vs Fitted")
}

check_heteroskedasticity(M_stepbest)
```

7.2.10 Stepwise Model Results

```
pred_step <- predict(M_stepbest, newdata = pollutants_after_VIF)
RMSE_step <- sqrt(mean((pollutants_after_VIF$length - exp(pred_step))^2))

pollutants_withX <- get_data(remove_indices = FALSE)
plot(pollutants_withX$X, pollutants$length, col = "black",
     xlab = "Observation Index",
     ylab = "Mean Leukocyte Telomere Length",
     main = "Actual and Predicted Outcome")
legend(x = "topright",
       legend = c("Actual outcome", "Predicted outcome"),
       col = c("black", "red"), pch = c(16, 16), bty = "n")
points(pollutants_withX$X, exp(pred_step), col = "red")
```

7.2.11 LASSO and Ridge Regression

```
# Extract covariates and outcome
y <- pollutants_after_VIF$length
# Take the log
y <- log(y)
# Split into test and train sets
ntrain <- 600
train_id <- 1:ntrain
# Obtain the design matrix from the full model
X <- model.matrix(Mfull_after_VIF)[,-1]
X_train <- X[train_id,]
```

```
X_test <- X[-train_id,]
y_train <- y[train_id]
y_test <- y[-train_id]

library(glmnet)
### LASSO
## fit LASSO with crossval
cvfit_lasso <-  cv.glmnet(x=X_train,y=unlist(y_train),alpha = 1)
## plot MSPEs by lambda
plot(cvfit_lasso)
coef(cvfit_lasso, s=’lambda.min’)
pred_lasso <- predict(cvfit_lasso, newx=X_test,  s="lambda.min")
## RMSE in test set
RMSE_lasso <- sqrt(mean((pred_lasso-y_test)^2))

### Ridge Regression
## fit Ridge Regression with crossval
cvfit_ridge <-  cv.glmnet(x=X_train,y=unlist(y_train),alpha = 1)
## plot MSPEs by lambda
plot(cvfit_ridge)
coef(cvfit_ridge, s=’lambda.min’)
pred_ridge <- predict(cvfit_ridge, newx=X_test,  s="lambda.min")
## RMSE in test set
RMSE_ridge <- sqrt(mean((pred_ridge-y_test)^2))
```

7.2.12 Leverage

```
check_leverage <- function(dataset) {
  M <- lm(length~., data=dataset)
  h <- hatvalues(M)
  ids <- which(h>2*(dim(model.matrix(M))[2])/nobs(M))
  # Plot
  par(mfrow = c(1,1))

  plot(h ,ylab="Leverage", main="Leverage of Observations")
  abline(h=2*mean(h),lty=2) ## add line at 2hbar
  points(h[ids]~ids,col="red",pch=19) ## add red points >2hbar
  text(x=ids,y=h[ids], labels=ids, cex= 0.6, pos=2) ## label points >2hbar
  return(ids)
}
high_leverage_ids <- check_leverage(pollutants_after_VIF)
length(high_leverage_ids)
```

7.2.13 Stepwise Model - Y-Outliers

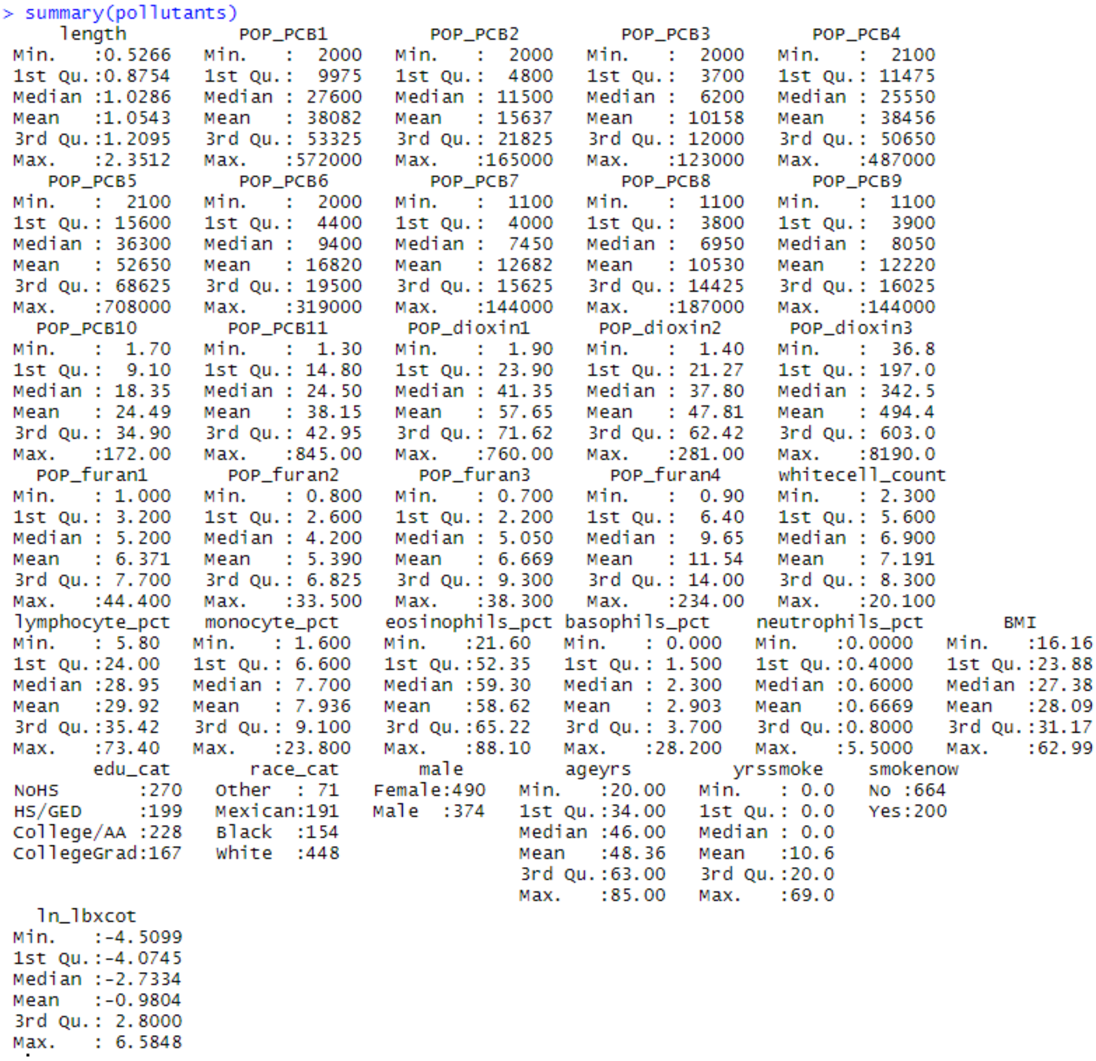
```
check_y_outliers <- function(Model, ids) {
  jack <- rstudent(Model)
  plot(abs(jack),ylab="|Studentized Jackknife Residuals|",
       main="Abs Studentized Jackknife Residuals for Observations")
  points(abs(jack)[ids]~ids,col="red",pch=19) ## add high leverage points
  text(ids,abs(jack)[ids], labels=ids, cex= 0.6, pos=2) ## label points >2hbar
}

check_y_outliers(M_stepbest, high_leverage_ids)
```

7.2.14 Stepwise Model - Checking Influence

```
plot_without_influential_pts <- function(pred, dataset, omit_ind, plotname) {
  ## Omit the supposedly influential points and fit the model
  M_omit <- lm(log(length) ~ ageyrs + POP_furan3, data = dataset[-omit_ind,])
  pred_omit <- predict(M_omit,newdata = dataset)

  ## plot different fitted values
  plot(log(dataset$length)~dataset$ageyrs,
       ylab="log Mean Leukocyte Telomere Length",xlab="Age (Years)",
       main=plotname)
  # fitted values based on the full data fit
  points(pred~dataset$ageyrs,col="blue",pch=19)
  # fitted values based on the data without the influential points
  points(pred_omit~dataset$ageyrs,col="red",pch=19)
  text(log(dataset$length)[omit_ind]~dataset$ageyrs[omit_ind],
       labels=omit_ind,pos=4)
  legend(x = "topright",
         legend = c("Regressing With Full Data",
                    "Regressing Without Influential Points"),
         col = c("blue", "red"), pch = c(16, 16), bty = "o")
}

# dataset must be unshuffled or else the index labels are incorrect
check_influence <- function(Model, dataset) {
  # number of covariates  and number of observations
  p <- dim(model.matrix(Model))[2] - 1
  n <- nobs(Model)

  # get fitted values based on entire sample
  pred <- predict(Model, newdata = dataset)

  ##----------------DFFITS-----------------
  dffits_m <- dffits(Model)

  ## plot DFFITS
  plot(dffits_m,ylab="DFFITS")
  abline(h=2*sqrt((p+1)/n),lty=2)  ## add thresholds
  abline(h=-2*sqrt((p+1)/n),lty=2)
  ## highlight influential points
  dff_ind <- which(abs(dffits_m)>2*sqrt((p+1)/n))
  ## add red points
  points(dffits_m[dff_ind]~dff_ind,col="red",pch=19)
  ## label high influence points
  text(y=dffits_m[dff_ind],x=dff_ind, labels=dff_ind, pos=2)

  plot_without_influential_pts(pred, dataset, dff_ind,
                               "Without Influential Points (DFFITS)")

  ##----------------Cook’s Distance--------------
  D <- cooks.distance(Model) # Cook’s distance
  ## influential points
  inf_ind <- which(D > 4/(864)) # use 4/N as the rule of thumb threshold
  ## plot cook’s Distance
  plot(D,ylab="Cook’s Distance")
  ## add red points
  points(D[inf_ind]~inf_ind,col="red",pch=19)
  ## label high influence points
  text(y=D[inf_ind],x=inf_ind, labels=inf_ind, pos=4)

  plot_without_influential_pts(pred, dataset, inf_ind,
                               "Without Influential Points (Cook’s Distance)")

  ##----------------DFBETAS--------------
  DFBETAS <- dfbetas(Model)
  dim(DFBETAS)
  ## beta1 (ageyrs)
  plot(DFBETAS[,2], type="h",xlab="Obs. Number",
       ylab=expression(paste("DFBETAS: ",beta[1])))
  show_points <- order( -abs(DFBETAS[,2]))[1:3]
  points(x=show_points,y=DFBETAS[show_points,2],pch=19,col="red")
  text(x=show_points,y=DFBETAS[show_points,2],labels=show_points,pos=2)

  ## beta2 (POP_furan3)
  plot(DFBETAS[,3], type="h",xlab="Obs. Number",
       ylab=expression(paste("DFBETAS: ",beta[2])))
  show_points <- order( -abs(DFBETAS[,3]))[1:3]
  points(x=show_points,y=DFBETAS[show_points,3],pch=19,col="red")
  text(x=show_points,y=DFBETAS[show_points,3],labels=show_points,pos=4)

  ## rule of thumb
  dfb_ind1 <- which(abs(DFBETAS[,2])>2/sqrt(n))
  dfb_ind2 <- which(abs(DFBETAS[,3])>2/sqrt(n))

  plot_without_influential_pts(pred, dataset, dfb_ind1,
                               "Without Influential Points (DFBETAS: beta1)")
  plot_without_influential_pts(pred, dataset, dfb_ind2,
                               "Without Influential Points (DFBETAS: beta2)")
}

# Analyze influence on unshuffled data to preserve indices
pollutants_unshuffled <- get_data(shuffle = FALSE)
Mfull_after_VIF_unshuffled <- eliminate_by_VIF(pollutants_unshuffled)
pollutants_after_VIF_unshuffled <-
  pollutants_unshuffled[, names(pollutants_unshuffled) %in%
                          c("length", "male", "edu_cat", "race_cat",
                          names(coef(Mfull_after_VIF_unshuffled)))]
M_stepbest_unshuffled <- lm(log(length) ~ ageyrs + POP_furan3,
                                  data = pollutants_after_VIF_unshuffled)
check_influence(M_stepbest_unshuffled, pollutants_after_VIF_unshuffled)
```

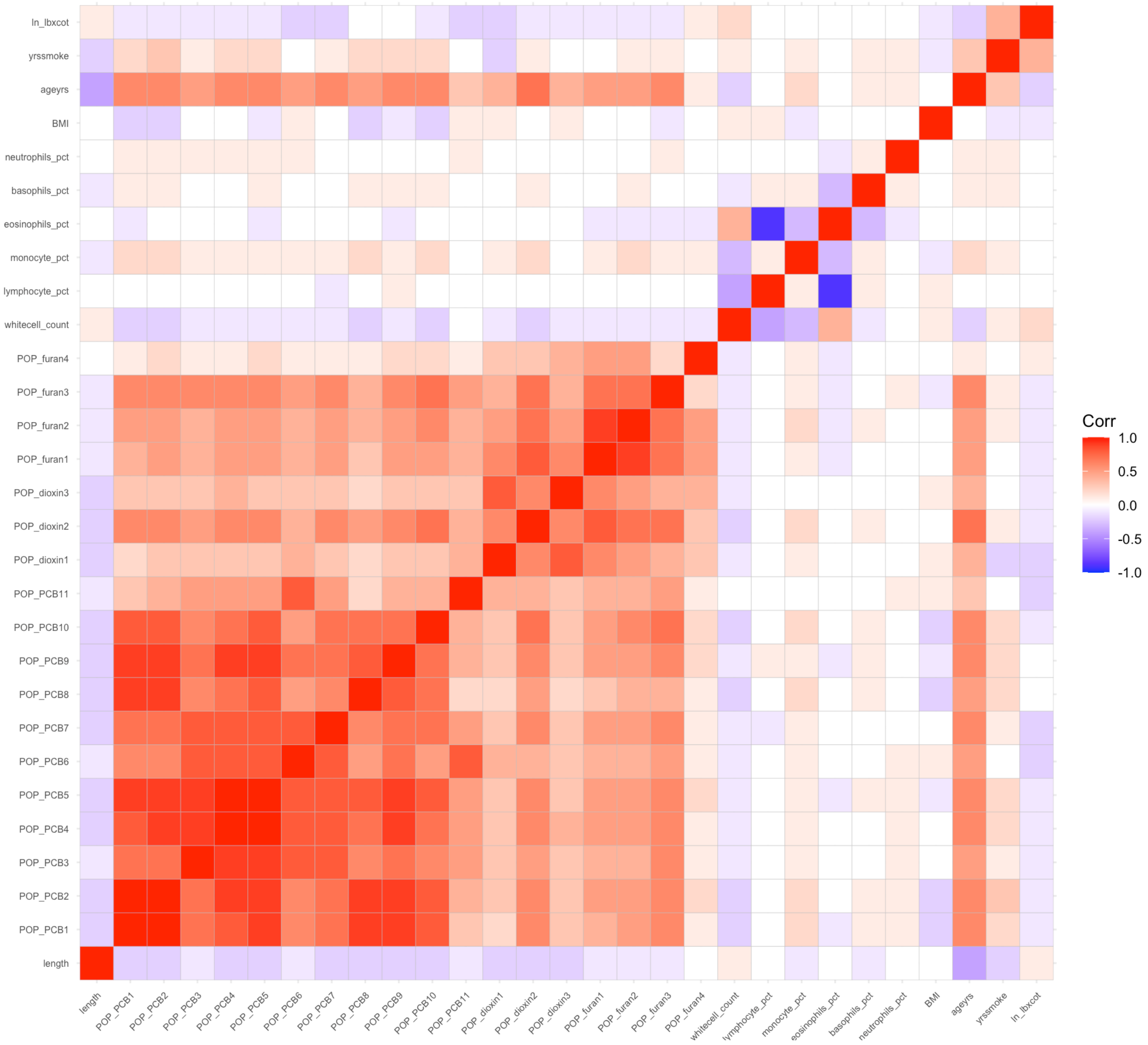
7.3 Plots and Pictures

7.3.1 Summary Statistics

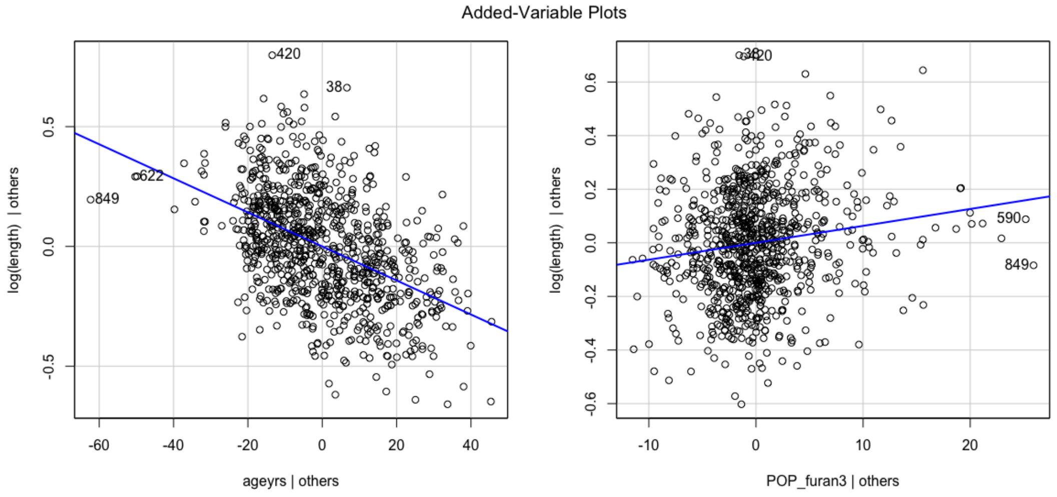


```
> summary(pollutants)
     length           POP_PCB1          POP_PCB2          POP_PCB3          POP_PCB4     
 Min.   :0.5266   Min.   :  2000   Min.   :  2000   Min.   :  2000   Min.   :  2100  
 1st Qu.:0.8754   1st Qu.:  9975   1st Qu.:  4800   1st Qu.:  3700   1st Qu.: 11475  
 Median :1.0286   Median : 27600   Median : 11500   Median :  6200   Median : 25550  
 Mean   :1.0543   Mean   : 38082   Mean   : 15637   Mean   : 10158   Mean   : 38456  
 3rd Qu.:1.2095   3rd Qu.: 53325   3rd Qu.: 21825   3rd Qu.: 12000   3rd Qu.: 50650  
 Max.   :2.3512   Max.   :572000   Max.   :165000   Max.   :123000   Max.   :487000  
    POP_PCB5          POP_PCB6          POP_PCB7          POP_PCB8          POP_PCB9     
 Min.   :  2100   Min.   :  2000   Min.   :  1100   Min.   :  1100   Min.   :  1100  
 1st Qu.: 15600   1st Qu.:  4400   1st Qu.:  4000   1st Qu.:  3800   1st Qu.:  3900  
 Median : 36300   Median :  9400   Median :  7450   Median :  6950   Median :  8050  
 Mean   : 52650   Mean   : 16820   Mean   : 12682   Mean   : 10530   Mean   : 12220  
 3rd Qu.: 68625   3rd Qu.: 19500   3rd Qu.: 15625   3rd Qu.: 14425   3rd Qu.: 16025  
 Max.   :708000   Max.   :319000   Max.   :144000   Max.   :187000   Max.   :144000  
   POP_PCB10         POP_PCB11       POP_dioxin1       POP_dioxin2      POP_dioxin3    
 Min.   :  1.70   Min.   :  1.30   Min.   :  1.90   Min.   :  1.40   Min.   :  36.8  
 1st Qu.:  9.10   1st Qu.: 14.80   1st Qu.: 23.90   1st Qu.: 21.27   1st Qu.: 197.0  
 Median : 18.35   Median : 24.50   Median : 41.35   Median : 37.80   Median : 342.5  
 Mean   : 24.49   Mean   : 38.15   Mean   : 57.65   Mean   : 47.81   Mean   : 494.4  
 3rd Qu.: 34.90   3rd Qu.: 42.95   3rd Qu.: 71.62   3rd Qu.: 62.42   3rd Qu.: 603.0  
 Max.   :172.00   Max.   :845.00   Max.   :760.00   Max.   :281.00   Max.   :8190.0  
   POP_furan1       POP_furan2       POP_furan3        POP_furan4     whitecell_count 
 Min.   : 1.000   Min.   : 0.800   Min.   : 0.700   Min.   :  0.90   Min.   : 2.300  
 1st Qu.: 3.200   1st Qu.: 2.600   1st Qu.: 2.200   1st Qu.:  6.40   1st Qu.: 5.600  
 Median : 5.200   Median : 4.200   Median : 5.050   Median :  9.65   Median : 6.900  
 Mean   : 6.371   Mean   : 5.390   Mean   : 6.669   Mean   : 11.54   Mean   : 7.191  
 3rd Qu.: 7.700   3rd Qu.: 6.825   3rd Qu.: 9.300   3rd Qu.: 14.00   3rd Qu.: 8.300  
 Max.   :44.400   Max.   :33.500   Max.   :38.300   Max.   :234.00   Max.   :20.100  
 lymphocyte_pct  monocyte_pct    eosinophils_pct basophils_pct    neutrophils_pct        BMI       
 Min.   : 5.80   Min.   : 1.600   Min.   :21.60   Min.   : 0.000   Min.   :0.0000   Min.   :16.16  
 1st Qu.:24.00   1st Qu.: 6.600   1st Qu.:52.35   1st Qu.: 1.500   1st Qu.:0.4000   1st Qu.:23.88  
 Median :28.95   Median : 7.700   Median :59.30   Median : 2.300   Median :0.6000   Median :27.38  
 Mean   :29.92   Mean   : 7.936   Mean   :58.62   Mean   : 2.903   Mean   :0.6669   Mean   :28.09  
 3rd Qu.:35.42   3rd Qu.: 9.100   3rd Qu.:65.22   3rd Qu.: 3.700   3rd Qu.:0.8000   3rd Qu.:31.17  
 Max.   :73.40   Max.   :23.800   Max.   :88.10   Max.   :28.200   Max.   :5.5000   Max.   :62.99  
        edu_cat        race_cat       male         ageyrs         yrssmoke     smokenow 
 NoHS       :270   Other   : 71   Female:490   Min.   :20.00   Min.   : 0.0   No :664  
 HS/GED     :199   Mexican:191   Male  :374   1st Qu.:34.00   1st Qu.: 0.0   Yes:200  
 College/AA :228   Black  :154               Median :46.00   Median : 0.0            
 CollegeGrad:167   White  :448               Mean   :48.36   Mean   :10.6            
                                               3rd Qu.:63.00   3rd Qu.:20.0            
                                               Max.   :85.00   Max.   :69.0            

   ln_lbxcot      
 Min.   :-4.5099  
 1st Qu.:-4.0745  
 Median :-2.7334  
 Mean   :-0.9804  
 3rd Qu.: 2.8000  
 Max.   : 6.5848  
```

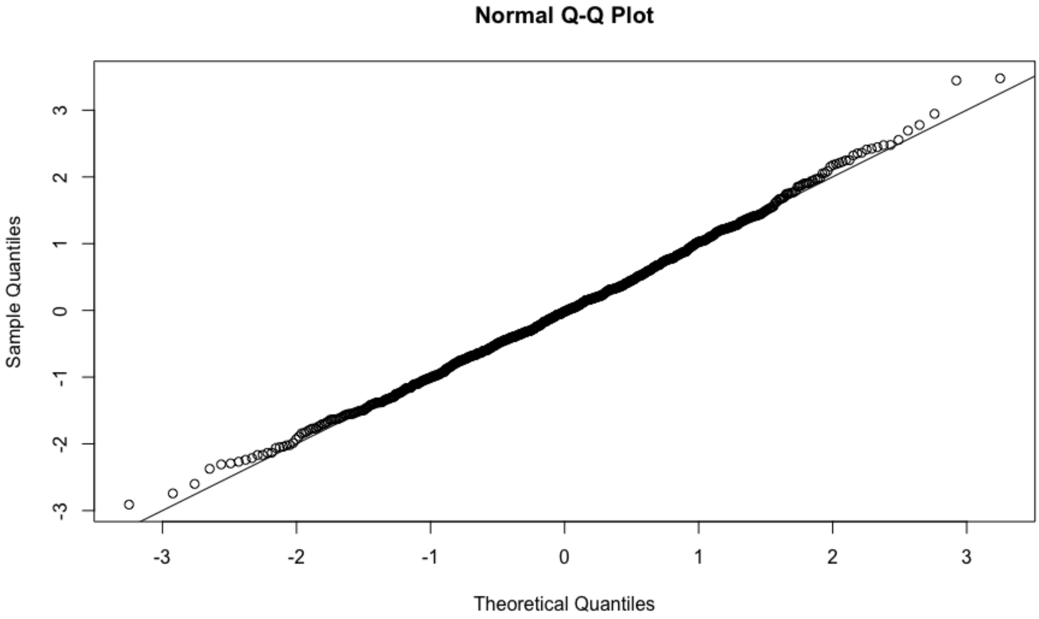
7.3.2 Correlation Matrix of all Covariates



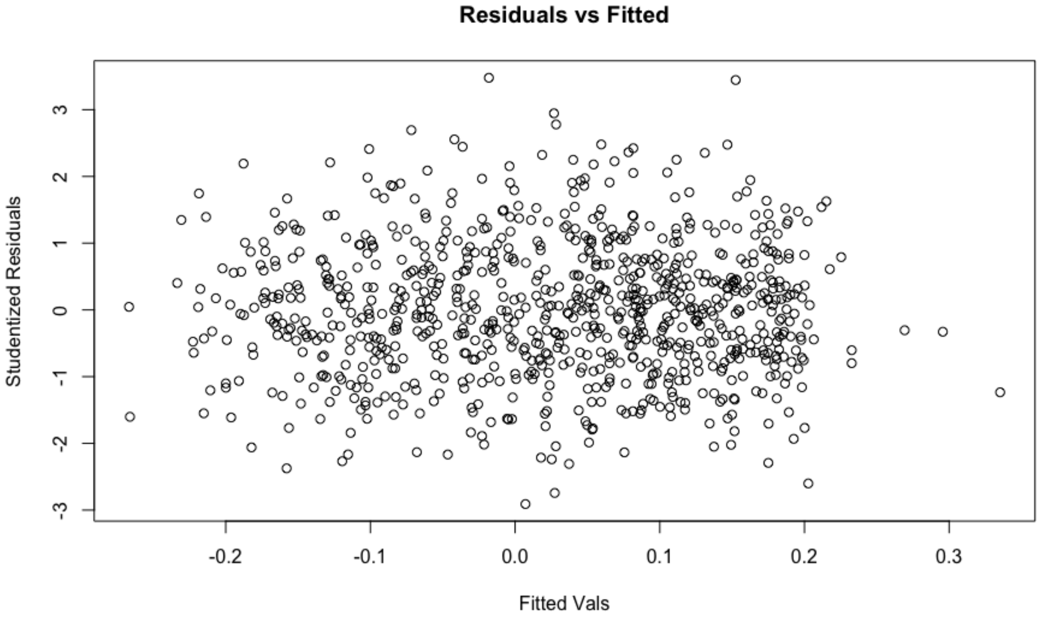
7.3.3 Stepwise Model - Checking Linearity (AvPlots)



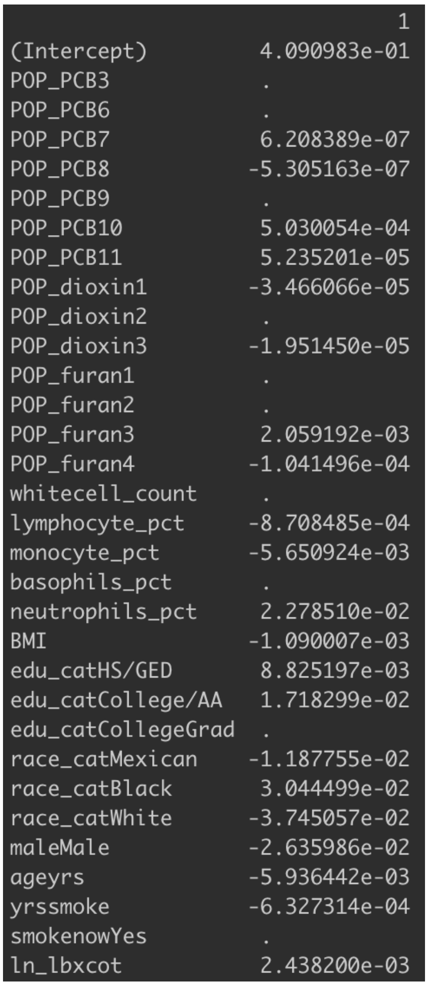
7.3.4 Stepwise Model - Checking Normality



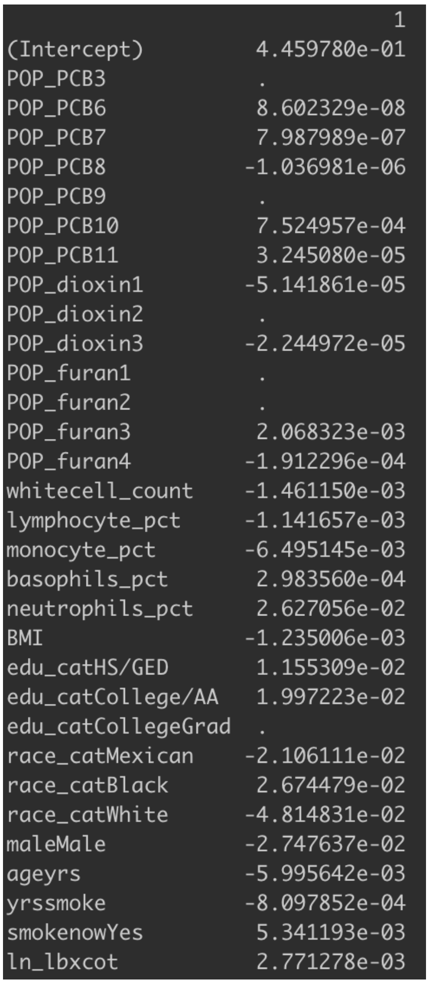
7.3.5 Stepwise Model - Checking Homoskedasticity



7.3.6 LASSO and Ridge Model - Coefficient Estimates

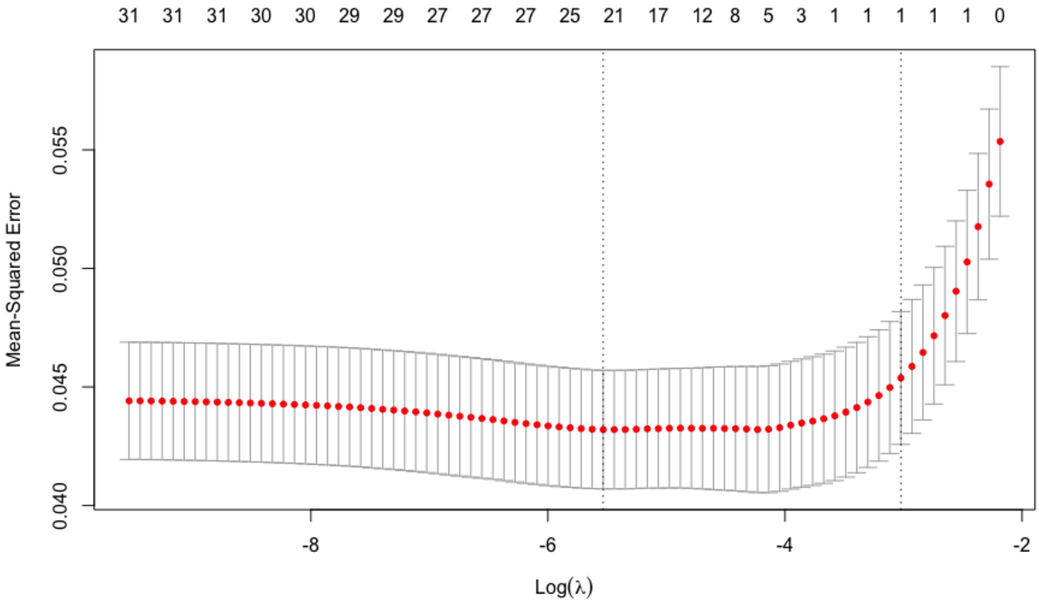


```
                                1
(Intercept)         4.090983e-01
POP_PCB3            .
POP_PCB6            .
POP_PCB7            6.208389e-07
POP_PCB8           -5.305163e-07
POP_PCB9            .
POP_PCB10           5.030054e-04
POP_PCB11           5.235201e-05
POP_dioxin1        -3.466066e-05
POP_dioxin2         .
POP_dioxin3        -1.951450e-05
POP_furan1          .
POP_furan2          .
POP_furan3          2.059192e-03
POP_furan4         -1.041496e-04
whitecell_count     .
lymphocyte_pct     -8.708485e-04
monocyte_pct       -5.650924e-03
basophils_pct       .
neutrophils_pct     2.278510e-02
BMI                -1.090007e-03
edu_catHS/GED       8.825197e-03
edu_catCollege/AA   1.718299e-02
edu_catCollegeGrad  .
race_catMexican    -1.187755e-02
race_catBlack       3.044499e-02
race_catWhite      -3.745057e-02
maleMale           -2.635986e-02
ageyrs             -5.936442e-03
yrssmoke           -6.327314e-04
smokenowYes         .
ln_lbxcot           2.438200e-03
```

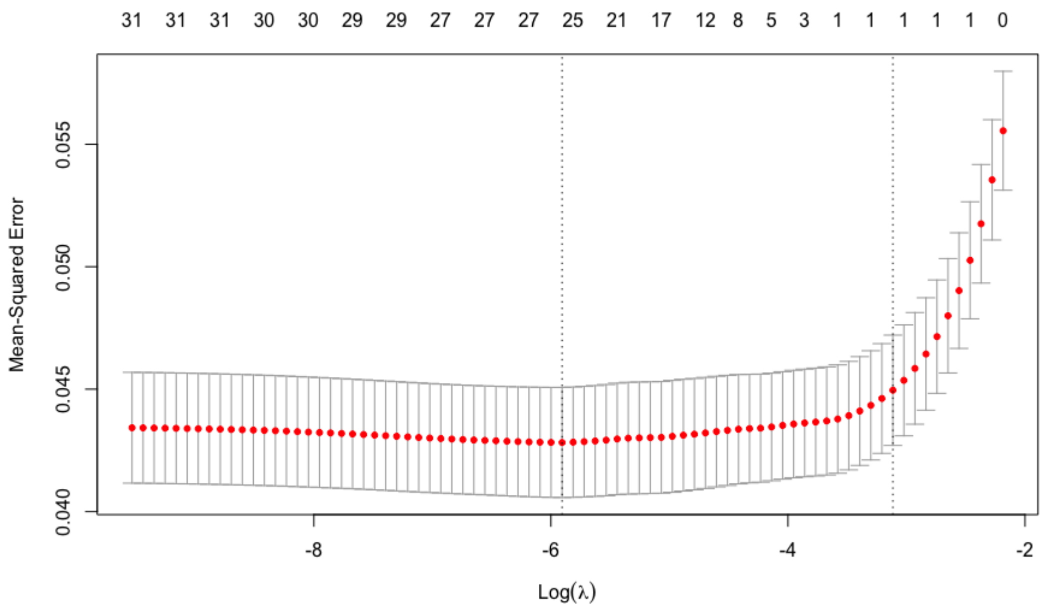


```
                                1
(Intercept)         4.459780e-01
POP_PCB3            .
POP_PCB6            8.602329e-08
POP_PCB7            7.987989e-07
POP_PCB8           -1.036981e-06
POP_PCB9            .
POP_PCB10           7.524957e-04
POP_PCB11           3.245080e-05
POP_dioxin1        -5.141861e-05
POP_dioxin2         .
POP_dioxin3        -2.244972e-05
POP_furan1          .
POP_furan2          .
POP_furan3          2.068323e-03
POP_furan4         -1.912296e-04
whitecell_count    -1.461150e-03
lymphocyte_pct     -1.141657e-03
monocyte_pct       -6.495145e-03
basophils_pct       2.983560e-04
neutrophils_pct     2.627056e-02
BMI                -1.235006e-03
edu_catHS/GED       1.155309e-02
edu_catCollege/AA   1.997223e-02
edu_catCollegeGrad  .
race_catMexican    -2.106111e-02
race_catBlack       2.674479e-02
race_catWhite      -4.814831e-02
maleMale           -2.747637e-02
ageyrs             -5.995642e-03
yrssmoke           -8.097852e-04
smokenowYes         5.341193e-03
ln_lbxcot           2.771278e-03
```

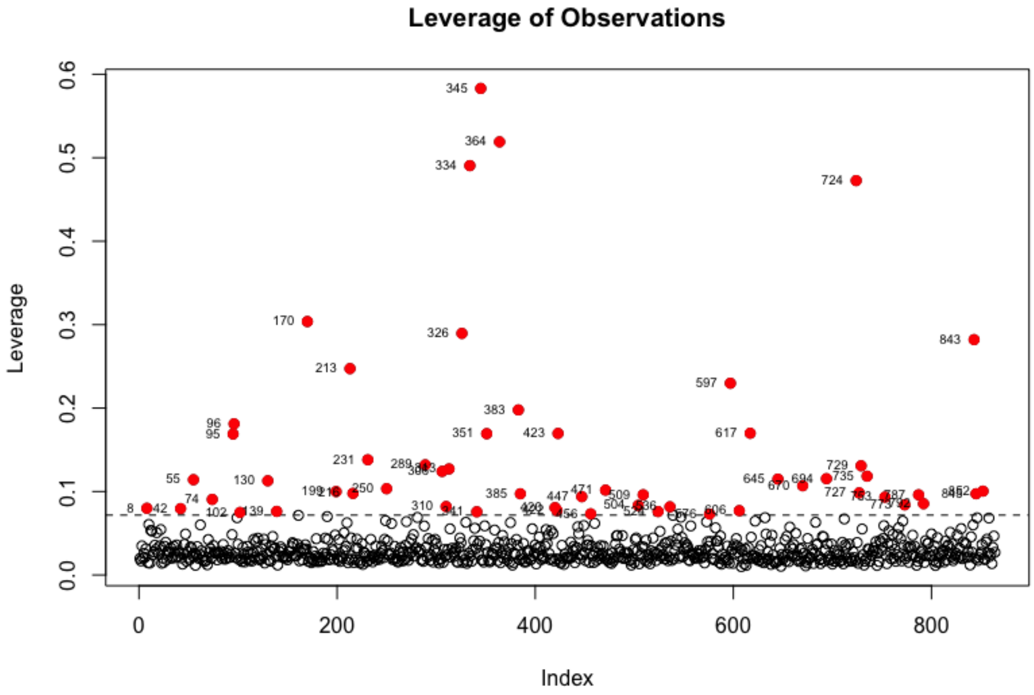
7.3.7 LASSO Model - Lambda Cross-Validation



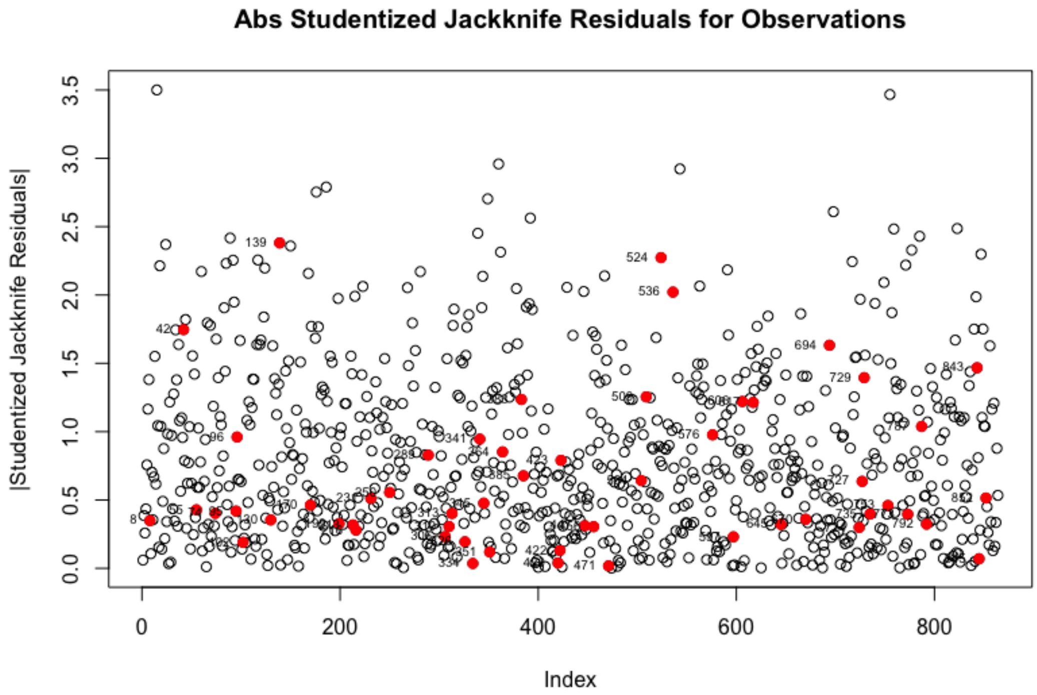
7.3.8 Ridge Regression Model - Lambda Cross-Validation



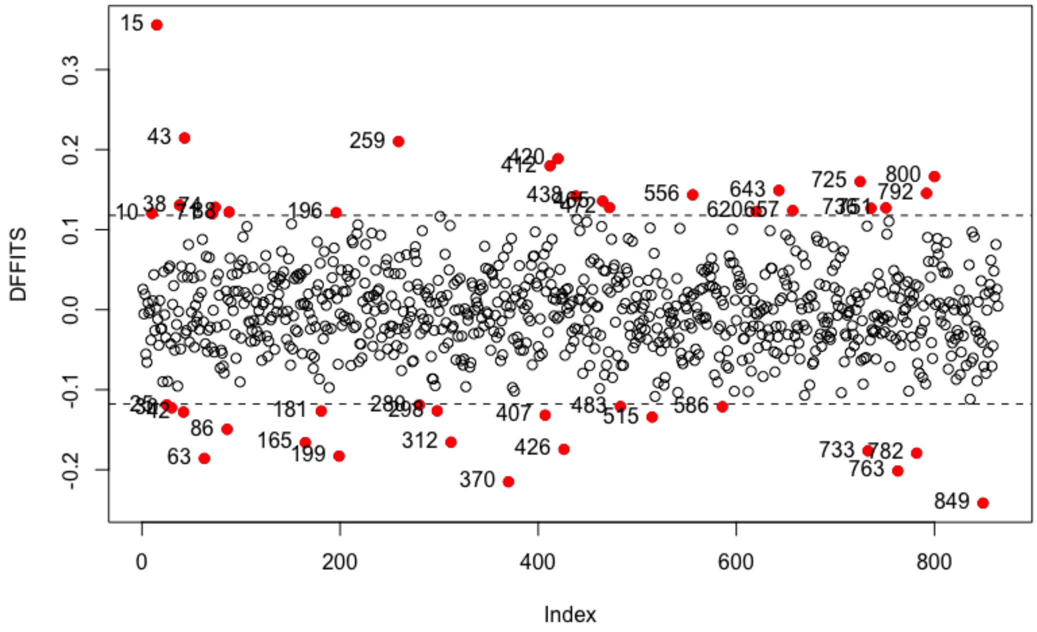
7.3.9 Leverage Plot

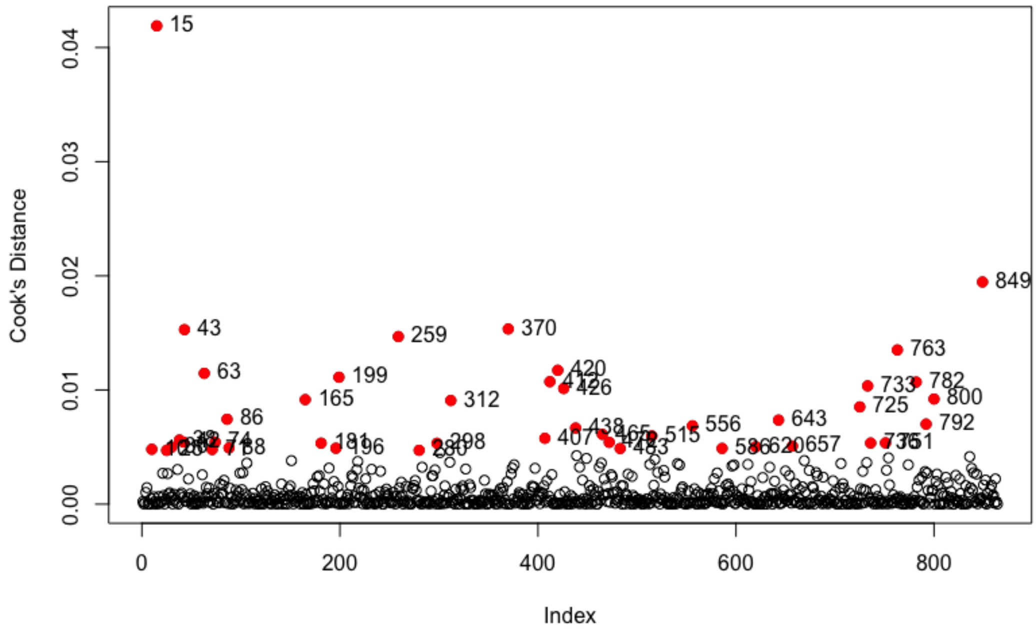


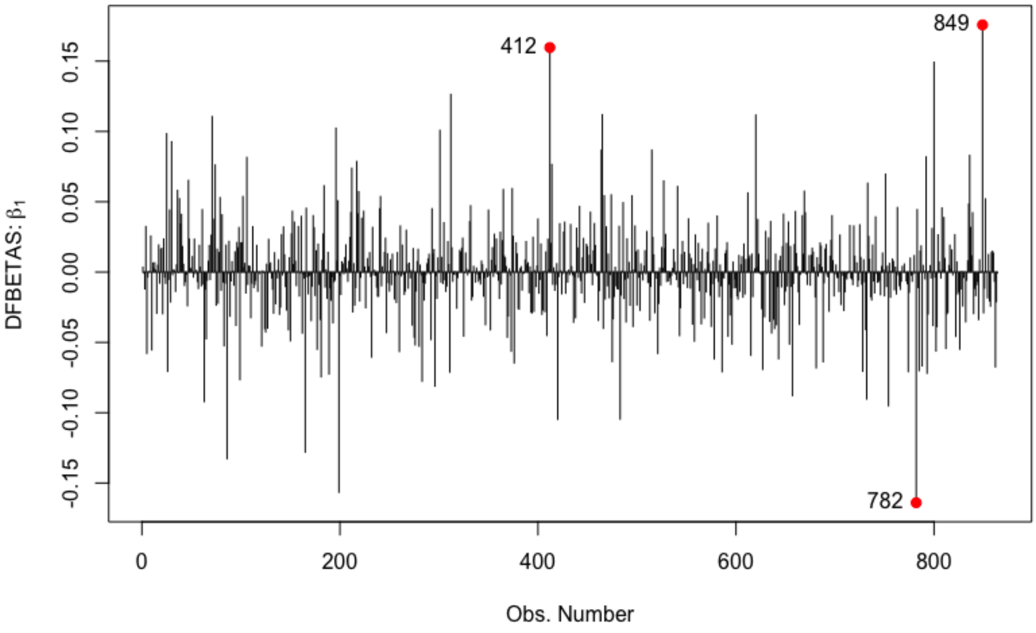
7.3.10 Stepwise Model - Y-Outliers Plot

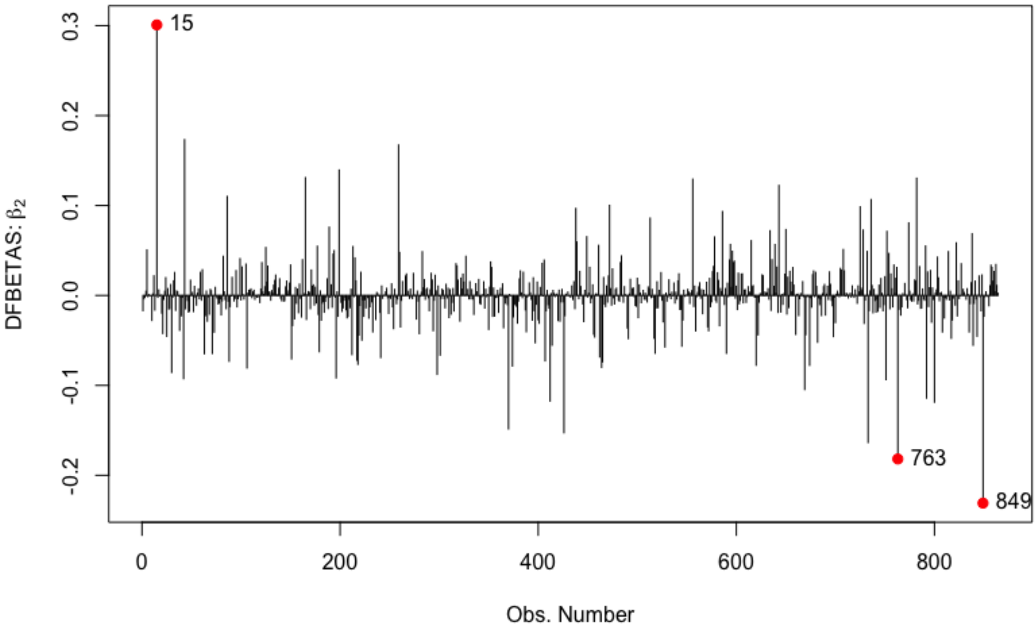


7.3.11 Stepwise Model - DFFITS, Cook's Distance, and DFBETAS

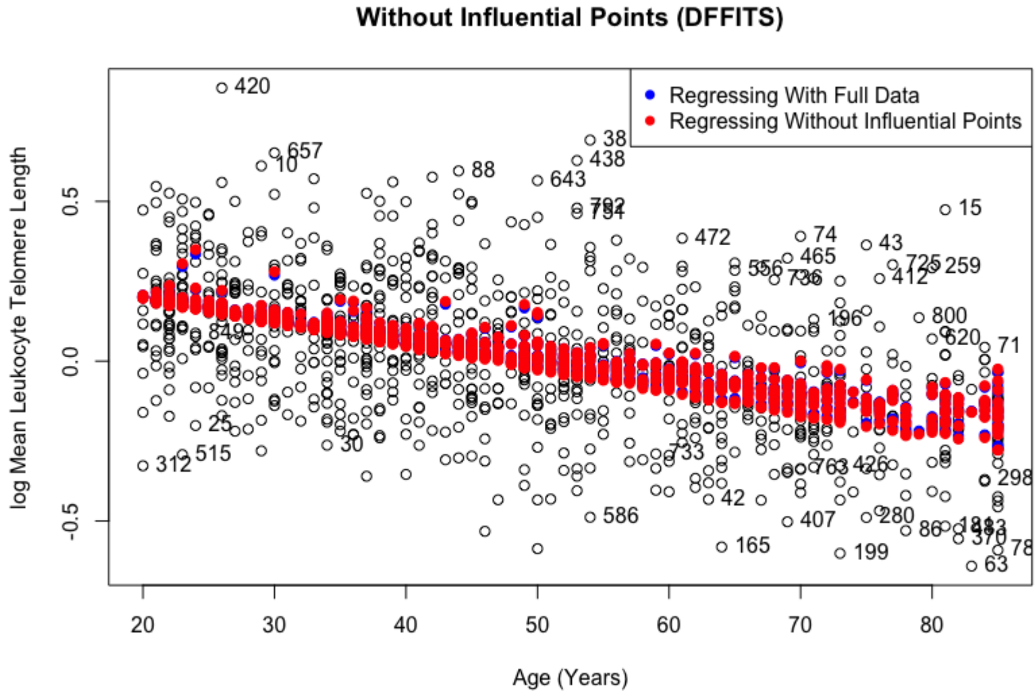


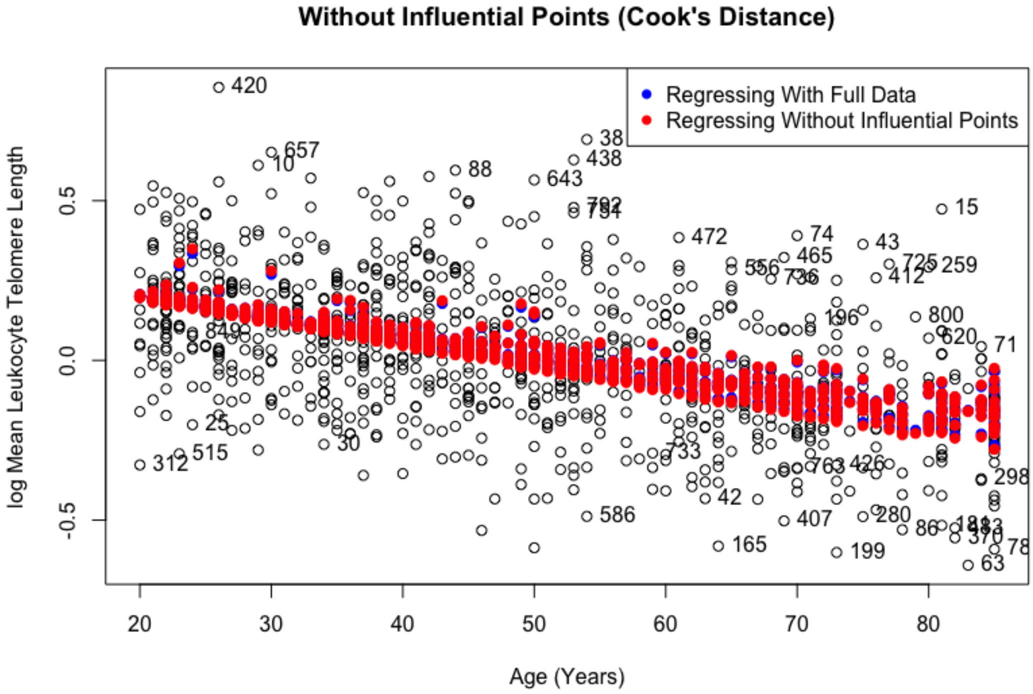

Cook's Distance
0.00
0.01
0.02
0.03
0.04
Index
0
200
400
600
800
15
43
63
86
74
88
165
181
196
199
259
280
298
312
370
407
412
420
426
438
465
483
515
556
586
620
643
657
725
733
735
751
763
782
792
800
849


DFBETAS: β_1
-0.15
-0.10
-0.05
0.00
0.05
0.10
0.15
Obs. Number
0
200
400
600
800
412
849
782

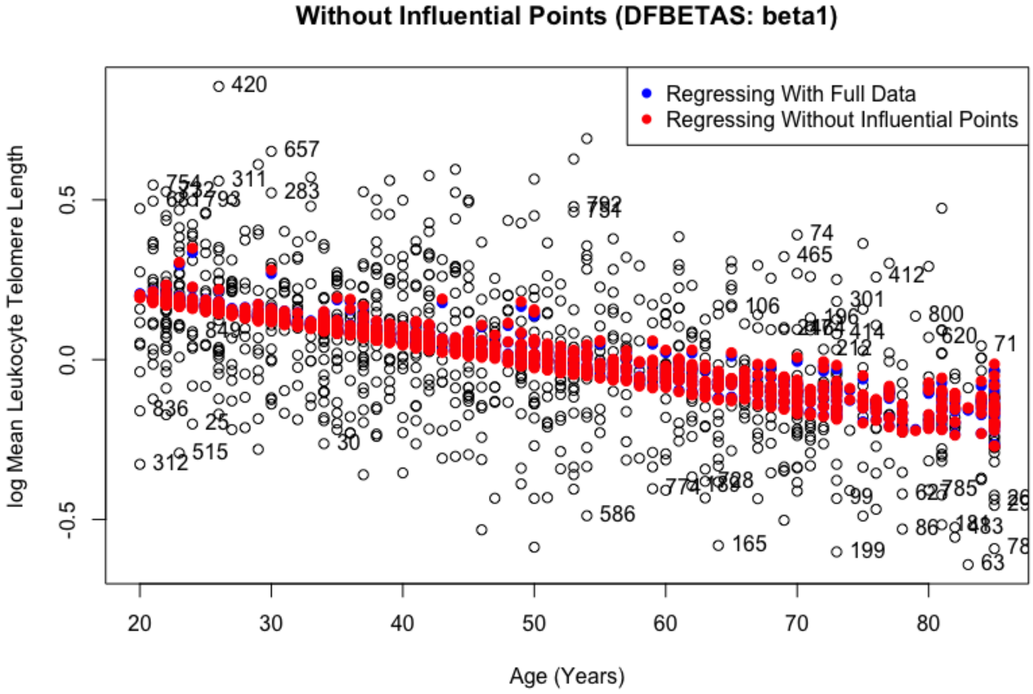


7.3.12 Comparing With And Without Influential Points

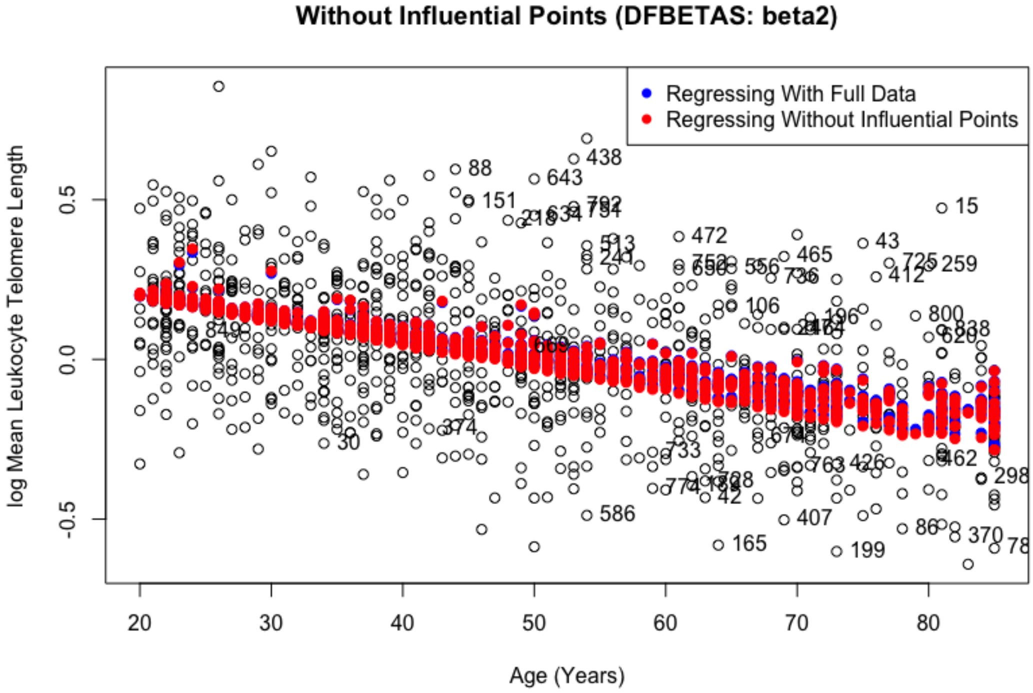




Without Influential Points (Cook's Distance)



Without Influential Points (DFBETAS: beta1)


Without Influential Points (DFBETAS: beta2)
Regressing With Full Data
Regressing Without Influential Points
log Mean Leukocyte Telomere Length
Age (Years)
-0.5
0.0
0.5
20
30
40
50
60
70
80
88
643
438
151
218
634
792
734
15
472
513
241
43
465
752
656
556
736
725
259
412
106
196
800
838
620
849
669
374
30
733
774
178
184
728
42
586
674
763
426
462
298
407
86
370
78
165
199