Assignment 2

Alexia Salomons, Nathan Maxwell Jones, Yauheniya Makarevich, group 71

15 March 2023

Exercise 1

a)

To investigate whether tree type influences total wood volume, we can perform a one-way ANOVA.

```
tree_df$type <- as.factor(tree_df$type)
tree_type_lm <- lm(volume~type, data=tree_df)
anova(tree_type_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##           Df Sum Sq Mean Sq F value Pr(>F)
## type       1    380     380     1.9   0.17
## Residuals 57  11395     200
```

With $p > 0.05$, we can conclude that *type* does not have a significant effect on *volume*. Because the factor *type* has two levels, we can apply an unpaired two sample t-test.

```
mask <- tree_df$type == "beech"
t.test(tree_df$volume[mask], tree_df$volume[!mask])
```

```
##
##  Welch Two Sample t-test
##
## data:  tree_df$volume[mask] and tree_df$volume[!mask]
## t = -1, df = 53, p-value = 0.2
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -12.33   2.17
## sample estimates:
## mean of x mean of y
##      30.2      35.2
```

Again we see that *type* is not significant. This supports the result from the ANOVA test. The estimated volume is 30.2 for Beech trees and 35.2 for Oak trees.

b)

To investigate this claim, we create two models, each including all three explanatory variables (*type*,

diameter and *height*). In the first model, we also include the pairwise interaction between *type* and *diameter*.

```
tree_type_d_lm <- lm(volume~height+type*diameter, data=tree_df)
anova(tree_type_d_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##               Df Sum Sq Mean Sq F value  Pr(>F)
## height         1   2188    2188  206.21 < 2e-16 ***
## type           1    431     431   40.65 4.2e-08 ***
## diameter       1   8577    8577  808.49 < 2e-16 ***
## type:diameter  1      6       6    0.52    0.47
## Residuals     54    573      11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the second model, we include the pairwise interaction between *type* and *height*.

```
tree_type_h_lm <- lm(volume~diameter+type*height, data=tree_df)
anova(tree_type_h_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##             Df Sum Sq Mean Sq F value  Pr(>F)
## diameter     1  10827   10827 1045.97 < 2e-16 ***
## type         1     45      45    4.37   0.041 *
## height       1    324     324   31.32 7.5e-07 ***
## type:height  1     19      19    1.88   0.176
## Residuals   54    559      10
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that both pairwise interactions are not significant. Therefore, we can conclude that both *height* and *diameter* have the same influence on *volume* regardless of *type*.

c)

In (b), we saw that the interactions of *height* and *diameter* with *type* were not significant, and so we will investigate a purely additive model (assuming no interactions).

```
tree_add_all_lm <- lm(volume~diameter+height+type, data=tree_df)
drop1(tree_add_all_lm, test= "F")
```

```
## Single term deletions
##
## Model:
## volume ~ diameter + height + type
##          Df Sum of Sq  RSS AIC F value  Pr(>F)
## <none>                 578 143
```

```
## diameter  1      8577 9155 304  815.61 < 2e-16 ***
## height    1       324  903 167   30.82 8.4e-07 ***
## type      1        23  602 143    2.21    0.14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the effect of *type* is not significant in the additive model. Therefore we will investigate an additive model that excludes *type*.

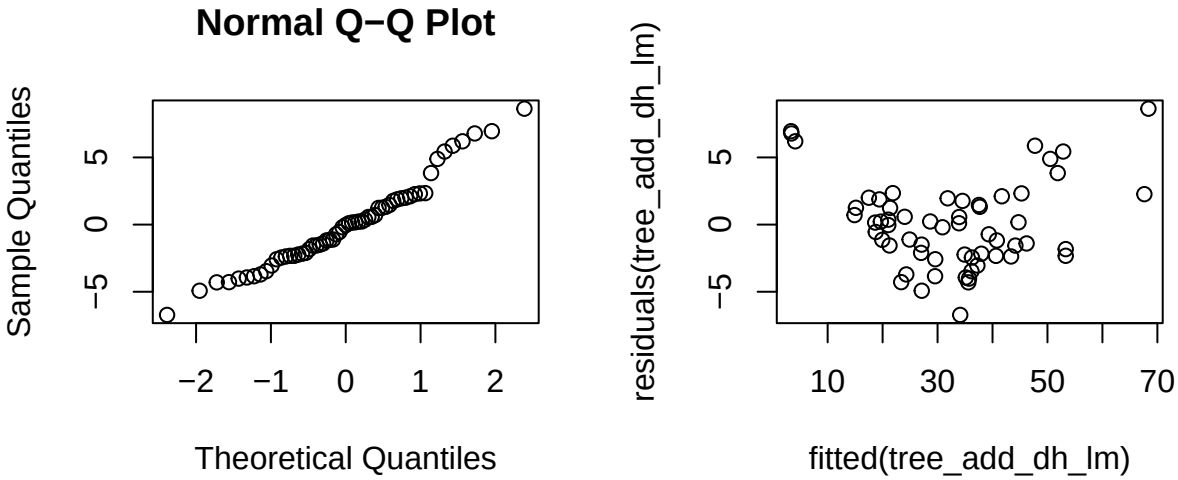
```
tree_add_dh_lm <- lm(volume~diameter+height, data=tree_df)
anova(tree_add_dh_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##           Df Sum Sq Mean Sq F value  Pr(>F)
## diameter   1  10827   10827  1007.8 < 2e-16 ***
## height     1    346     346    32.2 5.1e-07 ***
## Residuals 56    602      11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(tree_add_dh_lm)
```

```
##
## Call:
## lm(formula = volume ~ diameter + height, data = tree_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.724 -2.278 -0.034  1.820  8.629
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -64.3697     5.5577  -11.58  < 2e-16 ***
## diameter      4.6325     0.1602   28.92  < 2e-16 ***
## height        0.4289     0.0755    5.68  5.1e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.28 on 56 degrees of freedom
## Multiple R-squared:  0.949,  Adjusted R-squared:  0.947
## F-statistic:  520 on 2 and 56 DF,  p-value: <2e-16
```

This model has a high R-squared value while using fewer variables, all of which are significant. Since simpler models are generally preferred, this is our model of choice to make predictions. As a final test, we need to check this model's assumptions to ensure that the conclusions we draw from it are valid:



While these plots are not perfect, we believe the model assumptions to be valid.

Therefore, the effects of *type*, *diameter* and *height* can be summarized as follows:

- The tree *type* does not affect volume significantly.
- Looking at the coefficients, we see that increasing both height and diameter result in an increase in volume, with diameter having a bigger impact (with a gradient of 4.63 compared to *height's* 0.43). This makes sense given that we know volume is proportional to the square of the diameter.

To predict the volume for a tree with the overall average diameter and height, we can use the following linear regression model:

$$volume = -64.37 + 4.63 * diameter + 0.43 * height$$

```
mean_d <- mean(tree_df$diameter)
mean_h <- mean(tree_df$height)
means <-  data.frame(diameter=c(mean_d), height=c(mean_h))

predict(tree_add_dh_lm, means, interval = "confidence")
```

```
##    fit  lwr  upr
## 1 32.6 31.7 33.4
```

Therefore we expect the volume for such a tree to be 32.6, with a 95% CI of [31.7, 33.4].

d)

Assuming that a tree is roughly cylindrical, we expect that *volume* would be proportional to the *height* multiplied by the square of *diameter*. We perform this transformation and add it as a new column in the data frame. We could apply the true transformation, $V = h \times \pi(d/2)^2$, but this would just add unnecessary constants which would already be captured in the regression coefficients. We also will not include *type* because it was not significant.

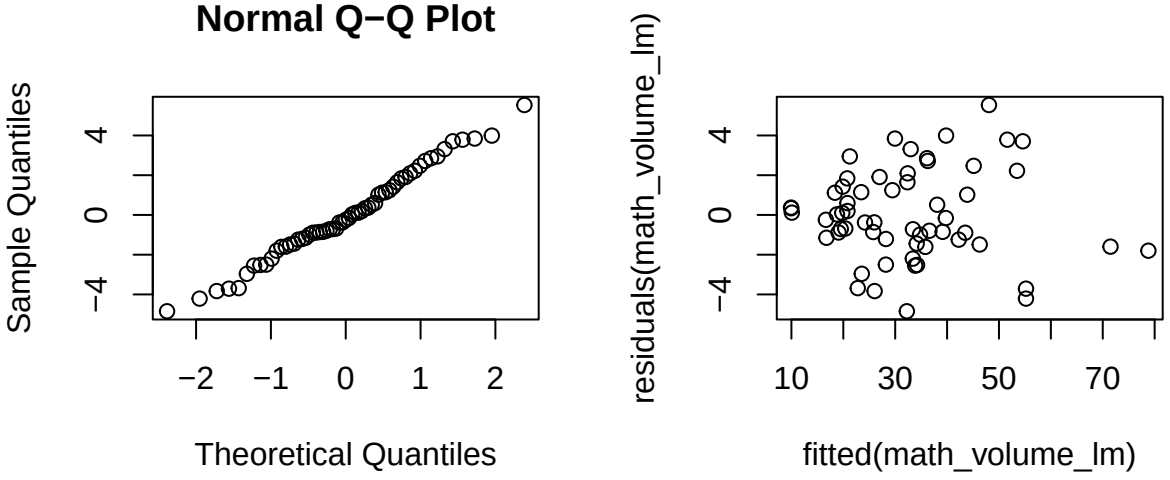
```
tree_df$math_volume <- tree_df$height * tree_df$diameter^2
math_volume_lm <- lm(volume~math_volume, data=tree_df)
anova(math_volume_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##             Df Sum Sq Mean Sq F value Pr(>F)
## math_volume  1  11477   11477    2201 <2e-16 ***
## Residuals   57    297       5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(math_volume_lm)
```

```
##
## Call:
## lm(formula = volume ~ math_volume, data = tree_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.846 -1.343 -0.245  1.533  5.532
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.79e-01   7.63e-01    -0.5     0.62
## math_volume  2.14e-03   4.57e-05    46.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.28 on 57 degrees of freedom
## Multiple R-squared:  0.975,  Adjusted R-squared:  0.974
## F-statistic: 2.2e+03 on 1 and 57 DF,  p-value: <2e-16
```

We see that this transformation does indeed produce an explanatory value with a significant effect. We also see that the R-squared (0.975) and adjusted R-squared (0.974) values are higher than that of the model chosen in (c) (tree_add_dh_lm), indicating that it better explains the data. Finally, we check the assumptions of this model.

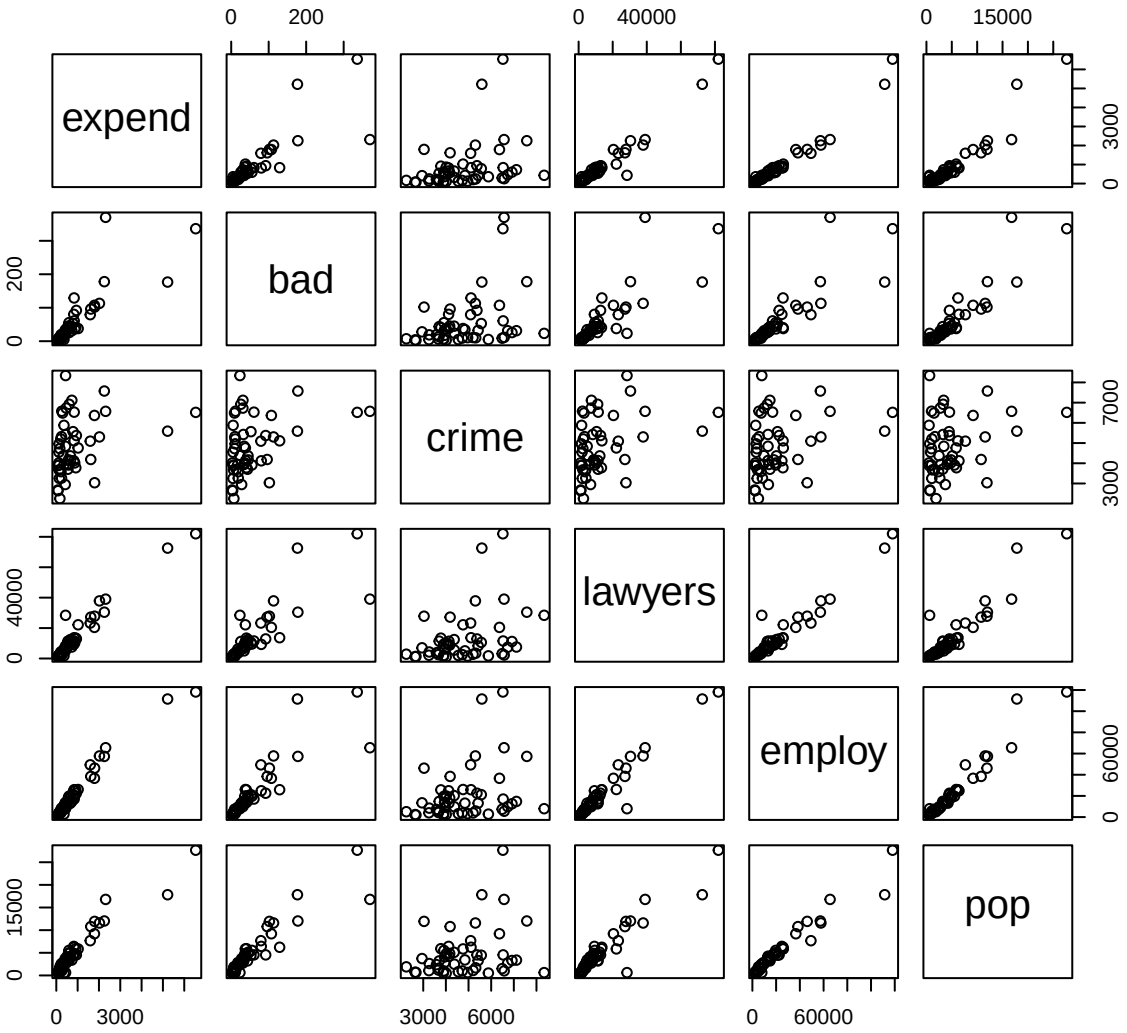


These plots are acceptable, meaning we can accept the model assumptions.

Exercise 2

a)

To investigate the interactions between all the variables of interest, we can plot the pairwise scatter plots for all their combinations:



We see that *expend*, our response variable, appears to have a positive correlation with all the explanatory variables except for *crime*. There appear to be several outliers at the high end of the data which could skew the model. We can also see that collinearity exists between the explanatory variables *bad*, *lawyers*, *employ* and *pop*. This is a problem since the redundant information will make the regression coefficients difficult to estimate.

We can use Cook's distance to find the influence points (a distance greater than 1 indicates an outlier)

```
crime_lm <- lm(expend~bad+crime+lawyers+employ+pop, data=crime_df)
cooks.distance(crime_lm)[cooks.distance(crime_lm) > 1]
```

```
##    5    8   35   44
## 4.91 3.51 1.09 2.70
```

We can see that indices of 5, 8, 35 and 44 are outliers, which we can remove:

```
crime_df_upd <- crime_df[-c(5,8,35,44),]
```

To further investigate collinearity, we can examine the correlations between all the explanatory variables, which confirms strong correlations between *bad*, *lawyers*, *employ* and *pop*.

```
round(cor(crime_df[, c(exp_vars)]), 2)
```

```
##          bad crime lawyers employ  pop
## bad     1.00  0.37    0.83   0.87 0.92
## crime   0.37  1.00    0.38   0.31 0.28
## lawyers 0.83  0.38    1.00   0.97 0.93
## employ  0.87  0.31    0.97   1.00 0.97
## pop     0.92  0.28    0.93   0.97 1.00
```

We can also use the VIF to see which variables are collinear (VIF > 5 is cause for concern).

```
vif(lm(expend~bad+crime+lawyers+employ+pop, data=crime_df))
```

```
##     bad   crime lawyers  employ     pop
##    8.36    1.49   16.97   33.59   32.94
```

This further confirms that collinearity exists for the variables *bad*, *lawyers*, *employ* and *pop*.

From this point on, we will proceed *without* the influence points.

b)

The step-up process was carried out. The variables added in order were *employ*, *crime* and *pop*, after which no further added variables had significant p-values. Hence the final model is as follows:

```
step_up_lm <- lm(expend~employ+crime+pop, data=crime_df_upd)
summary(step_up_lm)
```

```
##
## Call:
## lm(formula = expend ~ employ + crime + pop, data = crime_df_upd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -179.99  -49.64    0.48   51.19  266.63
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.47e+02   5.47e+01   -4.52  4.8e-05 ***
## employ       2.09e-02   3.95e-03    5.30  3.7e-06 ***
## crime        5.43e-02   1.13e-02    4.82  1.8e-05 ***
## pop          7.14e-02   1.79e-02    4.00  0.00025 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91.4 on 43 degrees of freedom
```

```
## Multiple R-squared:  0.974,  Adjusted R-squared:  0.973
## F-statistic:  547 on 3 and 43 DF,  p-value: <2e-16
```

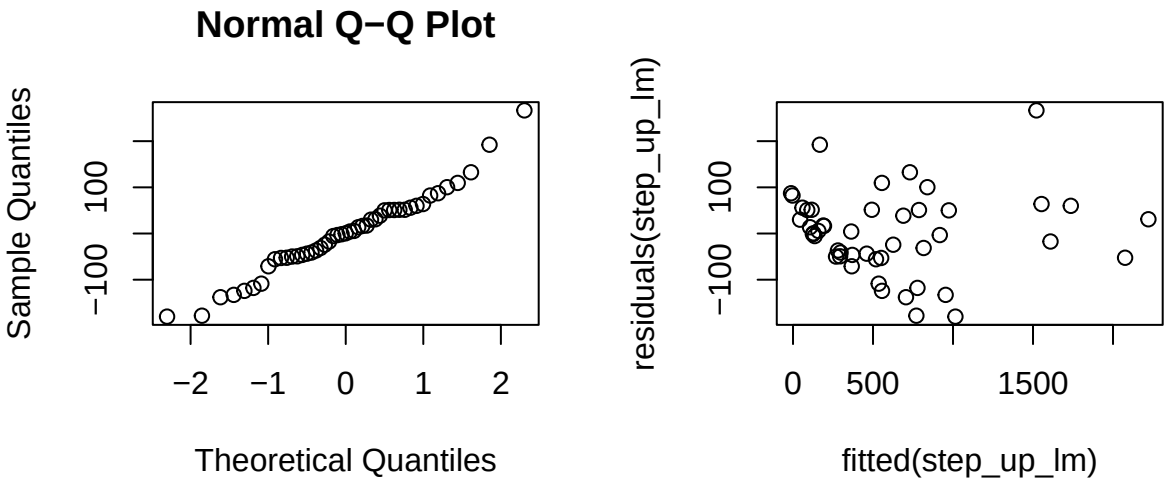
Final model: expend = -247 + 0.0209*employ + 0.0543*crime + 0.0714*pop ± error, with $R^2 = 0.974$. Using VIF we can again check the model for collinearity.

```
vif(step_up_lm)
```

```
## employ  crime    pop
##  18.08   1.14  17.51
```

We see that the step-up method failed to completely solve the problem of collinearity, since *employ* and *pop* have VIF > 5.

Finally, we check the model assumptions, which can be accepted based on the following plots:



c)

Using the step-up model found in (b), the 95% prediction interval for *expend* is given by:

```
new_data <- data.frame(bad=50, crime=5000, lawyers=5000, employ=5000, pop=5000)
predict(step_up_lm, new_data, interval="prediction", level=0.95)
```

```
##   fit lwr upr
## 1 486 258 713
```

We cannot improve this interval since we have already removed the influence points from the data.

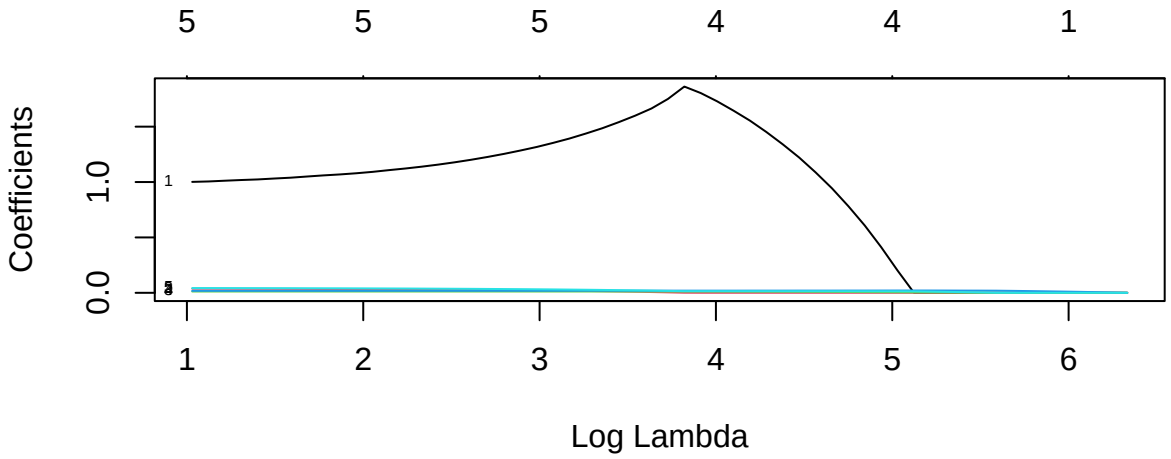
d)

We can apply the LASSO method as follows:

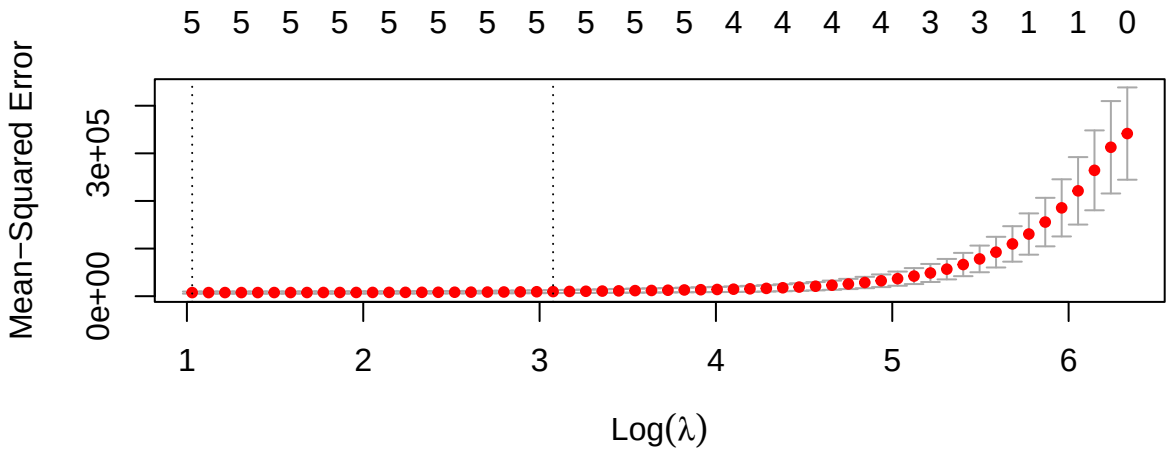
```
x <- as.matrix(crime_df_upd[, exp_vars])
y <- as.matrix(crime_df_upd[, c(response)])

# train-test splitting
train <- (sample(1:nrow(x), 0.67*nrow(x))) # train by using 2/3 of the data
x.train <- x[train,]; y.train <- y[train]
x.test <- x[-train,]; y.test <- y[-train]
```

```
# fitting the model
lasso.mod <- glmnet(x.train, y.train, alpha=1)
cv.lasso <- cv.glmnet(x.train,y.train,alpha=1,type.measure='mse')
```

```
plot(lasso.mod, label=T, xvar="lambda")
```




```
plot(cv.lasso)
```



The resulting choice of lambda is *lambda.1se*, which is given by:

```
(lambda.1se <- cv.lasso$lambda.1se)
```

```
## [1] 21.7
```

We can evaluate the resulting model by examining the Mean Squared Error (MSE) on the test set.

```
y.pred <- predict(lasso.mod, s=lambda.1se, newx=x.test)
mse.lasso <- mean((y.test - y.pred)^2); mse.lasso
```

```
## [1] 17262
```

To compare this to the step-up model in (b), we can find the MSE for this model on the same test set.

```
new_data <- data.frame(x.test)
y.pred <- predict(step_up_lm, new_data, interval="confidence", level=0.95)
mse.step_up <- mean((y.test - y.pred)^2); mse.step_up
```

```
## [1] 15212
```

We see that the step-up model outperforms the LASSO model, producing a smaller MSE. This could be because LASSO is better suited to situations with many more explanatory variables. It could also be because the step-up model was created based on the entire dataset which includes the test set, unlike the LASSO model, giving it an unfair advantage.

Exercise 3

a)

Studying the Data

We can start by looking at the summary data for each column:

```
titanic_df$PClass <- as.factor(titanic_df$PClass)
titanic_df$Sex <- as.factor(titanic_df$Sex)
summary(titanic_df)
```

```
##      Name           PClass          Age           Sex         Survived
##  Length:1313        1st:322   Min.   : 0    female:462   Min.   :0.000
##  Class :character   2nd:280   1st Qu.:21    male  :851   1st Qu.:0.000
##  Mode  :character   3rd:711   Median :28                 Median :0.000
##                               Mean   :30                 Mean   :0.343
##                               3rd Qu.:39                 3rd Qu.:1.000
##                               Max.   :71                 Max.   :1.000
##                               NA's   :557
```

We see that there were more people in 3rd class than in 1st and 2nd combined. Half the people were between the ages 21 and 39, with an average of 30 years old. Males made up 65% of the passengers, and 34% of all the passengers survived.

We can also examine how the sexes were distributed over the classes:

```
tot_comb <- xtabs(~PClass+Sex, data=titanic_df)
tot_comb
```

```
##       Sex
## PClass female male
##    1st    143  179
##    2nd    107  173
##    3rd    212  499
```

As expected, there are more males in each category, with the biggest difference in 3rd class.

We can examine the same distribution in terms of percentage of people who survived for each combination:

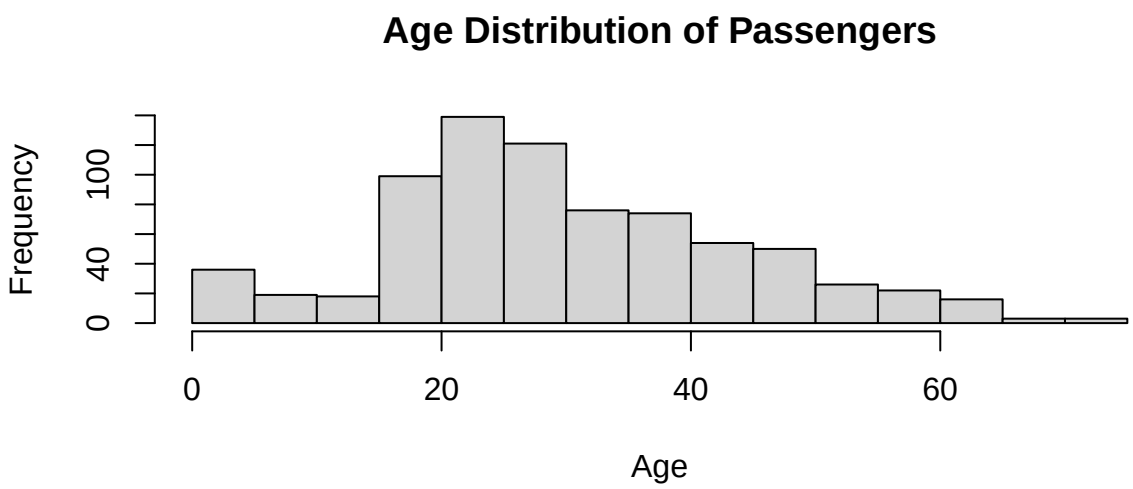
```
tot_comb.surv <- xtabs(Survived~PClass+Sex, data=titanic_df)
round(tot_comb.surv/tot_comb, 2)
```

```
##       Sex
## PClass female male
##    1st   0.94 0.33
##    2nd   0.88 0.14
##    3rd   0.38 0.12
```

It appears that the survival rate increases from 3rd to 1st class, and is much higher for females in all three cases, although this margin is less pronounced for 3rd class.

Finally, we can plot the distribution of ages:

```
hist(titanic_df$Age, xlab="Age", main="Age Distribution of Passengers")
```



We can see that the bulk of the ages are grouped around 25, with exponentially fewer people for older groupings.

Fitting the Model

First, we will removing rows with missing ages:

```
titanic_df_upd <- na.omit(titanic_df)
```

Next, we can fit a logistic regression model and test it as follows:

```
titanic_df_upd$PClass <- as.factor(titanic_df_upd$PClass)
titanic_df_upd$Sex <- as.factor(titanic_df_upd$Sex)
additive_lm <- glm(Survived ~ Age+PClass+Sex, data = titanic_df_upd, family = binomial)
```

```
drop1(additive_lm, test = "Chisq")
```

```
## Single term deletions
##
## Model:
## Survived ~ Age + PClass + Sex
##        Df Deviance AIC   LRT Pr(>Chi)
```

```
## <none>         695 705
## Age     1      724 732  28.5  9.6e-08 ***
## PClass  2      796 802 100.4  < 2e-16 ***
## Sex     1      910 918 214.8  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that all three variables have a significant effect on the odds of survival. To further interpret the results, we can look at the model summary and its coefficients.

```
summary(additive_lm)
```

```
##
## Call:
## glm(formula = Survived ~ Age + PClass + Sex, family = binomial,
##     data = titanic_df_upd)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.723  -0.707  -0.392   0.649   2.529
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.75966    0.39757    9.46  < 2e-16 ***
## Age         -0.03918    0.00762   -5.14  2.7e-07 ***
## PClass2nd   -1.29196    0.26008   -4.97  6.8e-07 ***
## PClass3rd   -2.52142    0.27666   -9.11  < 2e-16 ***
## Sexmale     -2.63136    0.20151  -13.06  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1025.57  on 755  degrees of freedom
## Residual deviance:  695.14  on 751  degrees of freedom
## AIC: 705.1
##
## Number of Fisher Scoring iterations: 5
```

Thus, the estimated odds is:

$$\hat{o}_k = \frac{\widehat{P(Y_k = 1)}}{\widehat{P(Y_k = 0)}} \approx e^{3.76 - 0.04 * Age_k - 1.29 * PClass2nd_k - 2.52 * PClass3rd_k - 2.63 * Sexmale_k}$$

We see that all the coefficients are negative. This means that increasing age, being male and moving down in class all lower a passenger's odds of survival. Moreover, the magnitude of the coefficients indicates how sensitive these odds are to each variable. From this we can infer that 3rd class has worse odds than 2nd class, and that being a male is the most influential factor in decreasing one's

odds of survival. While increasing age also lowers one's odds, it has the smallest effect compared to to the other two variables.

b)

Investigating the interaction between *Age* and *PClass*:

```
anova(glm(Survived ~ Age*PClass, data = titanic_df_upd, family = binomial), test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Survived
##
## Terms added sequentially (first to last)
##
##
##             Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                          755       1026
## Age          1      2.8       754       1023    0.091 .
## PClass       2    112.8       752        910   <2e-16 ***
## Age:PClass   2      1.2       750        909    0.558
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the interaction is not siginficant. Next, we investigating the interaction between *Age* and *Sex*:

```
anova(glm(Survived ~ Age*Sex, data = titanic_df_upd, family = binomial), test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Survived
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                       755       1026
## Age       1      2.8       754       1023    0.091 .
## Sex       1    227.1       753        796  < 2e-16 ***
## Age:Sex   1     25.0       752        771  5.6e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here, the interaction is significant, and so we wish to include it in our model. *PClass* was significant by itself, so we include it in the final model as follows:

```
class_and_interaction_lm <- glm(Survived ~ PClass+Age*Sex, data = titanic_df_upd, family = bino
summary(class_and_interaction_lm, test="Chisq")
```

```
##
## Call:
## glm(formula = Survived ~ PClass + Age * Sex, family = binomial,
##     data = titanic_df_upd)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.435  -0.656  -0.353   0.696   2.728
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.75656    0.43764    6.30  3.0e-10 ***
## PClass2nd   -1.54337    0.28736   -5.37  7.8e-08 ***
## PClass3rd   -2.65398    0.29142   -9.11  < 2e-16 ***
## Age          0.00244    0.01141    0.21     0.83
## Sexmale     -0.50819    0.44251   -1.15     0.25
## Age:Sexmale -0.07559    0.01501   -5.04  4.7e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1025.57  on 755  degrees of freedom
## Residual deviance:  667.08  on 750  degrees of freedom
## AIC: 679.1
##
## Number of Fisher Scoring iterations: 5
```

We see that *Age* and *Sexmale* are no longer significant, and the the AIC for this model (679.1) is lower than that of the purely additive model (705.1). Since we prefer smaller models with only significant variables, we choose the additive model as the final model.

Using this model we can predict the estimate for the probability of survival for each combination of levels of the factors *PClass* and *Sex* for a person of *Age* 55:

```
newdata <- data.frame(Age=c(55, 55, 55, 55, 55, 55), PClass=c("1st", "1st", "2nd", "2nd", "3rd"
predict(additive_lm, newdata, type="response")
```

```
##      1      2      3      4      5      6
## 0.8327 0.2638 0.5776 0.0896 0.2857 0.0280
```

Again we see that being a women appears to be the dominant factor in increased survival, and that the probability of survival also decreases as you go down in class (from 1st to 3rd)

c)

We can predict the survival status of the individuals by looking at the fitted values produced by the model, which gives the probability of survival. We can then set a threshold of 0.5. We can predict

that those with a probability lower than this did not survive, and those with a probability higher than this did survive. To measure the quality of this prediction method, we can use tools such as a confusion matrix and log likelihood as quality measures.

d)

Survived vs Sex

For 2x2 tables we can obtain exact p-value using the Fisher test as follows:

```
fisher.test(x=titanic_df_upd$Survived, y=titanic_df_upd$Sex)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  titanic_df_upd$Survived and titanic_df_upd$Sex
## p-value <2e-16
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.0586 0.1215
## sample estimates:
## odds ratio
##     0.0848
```

Therefore *Sex* and *Survived* are not independent.

Survived vs PClass

Since we have three classes, we must use the Chi-squared test to investigate the effect of passenger class:

```
chisq.test(x=titanic_df_upd$Survived, y=titanic_df_upd$PClass)
```

```
##
##  Pearson's Chi-squared test
##
## data:  titanic_df_upd$Survived and titanic_df_upd$PClass
## X-squared = 76, df = 2, p-value <2e-16
```

Therefore *PClass* and *Survived* are not independent. Since R does not issue a warning, we can assume that the requirement of at least 80% of the expected values being at least 5 has been met.

e)

No, the approach in (d) is not wrong. Rather there are advantages and disadvantages associated with both methods:

Logistic Regression

Advantages - Can handle a combination of continuous and categorical explanatory variables. Gives you detailed information about the role of explanatory variables since it tells you how the variable affects the response (sign of the coefficient), as well as how sensitive the response is to changes in the variable (magnitude of the coefficient).
Disadvantages - It is not well suited to predicting continuous response variables, and if the sample size is too small, it can suffer from overfitting.

vs

Contingency Table.

Advantages - Works for tables larger than 2x2. Works best for categorical explanatory variables.
Disadvantages - Does not work for continuous variables. Does not provide an exact p-value, only an approximation. Only tells you whether dependence exists, not the nature of the dependency. You also cannot make predictions using this method.

vs

Fisher

The Fisher test has the same advantages and disadvantages as the Contigency Table except for the following:

Advantages - Provides an exact p-value.
Disadvantages - Only works for 2x2.

Exercise 4

We will consider *pollib* as a factor, and the remaining variables as continuous.

```
coups_df$pollib <- as.factor(coups_df$pollib)
```

a)

Fitting the model using Poisson regression:

```
poison_glm <- glm(miltcoup ~ oligarchy + parties + pctvote + popn + size + numelec + numregim +
drop1(poison_glm, test= "Chisq")
```

```
## Single term deletions
##
## Model:
## miltcoup ~ oligarchy + parties + pctvote + popn + size + numelec +
##     numregim + pollib
##           Df Deviance AIC  LRT Pr(>Chi)
## <none>           28.2 113
## oligarchy  1     32.4 115 4.10   0.0428 *
## parties    1     35.3 118 7.06   0.0079 **
## pctvote    1     30.6 113 2.32   0.1275
## popn       1     30.6 113 2.35   0.1252
## size       1     29.2 112 0.99   0.3202
## numelec    1     28.4 111 0.18   0.6705
## numregim   1     29.1 112 0.81   0.3681
## pollib     2     35.6 116 7.33   0.0256 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(poison_glm)
```

```
##
## Call:
```

```
## glm(formula = miltcoup ~ oligarchy + parties + pctvote + popn +
##     size + numelec + numregim + pollib, family = poisson, data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.508  -0.953  -0.310   0.486   1.646
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.233427   0.997611   -0.23   0.8150
## oligarchy    0.072566   0.035346    2.05   0.0401 *
## parties      0.031221   0.011166    2.80   0.0052 **
## pctvote      0.015441   0.010103    1.53   0.1264
## popn         0.010959   0.007149    1.53   0.1253
## size        -0.000265   0.000269   -0.99   0.3244
## numelec     -0.029619   0.069625   -0.43   0.6705
## numregim     0.210943   0.233933    0.90   0.3672
## pollib1     -1.103244   0.655811   -1.68   0.0925 .
## pollib2     -1.690306   0.676650   -2.50   0.0125 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 65.945  on 35  degrees of freedom
## Residual deviance: 28.249  on 26  degrees of freedom
## AIC: 113.1
##
## Number of Fisher Scoring iterations: 5
```

We see that the variables that are significant in predicting the number of successful military coups are: *oligarchy*, *pollib*, and *parties*.

b)

The step down method was applied, removing the variables: *numelec*, *size*, *pctvote*, and *popn*, respectively. After which, all remaining variables were significant, resulting in the model:

```
final_plm <- glm(miltcoup ~ oligarchy + parties + pollib, data = coups_df, family = poisson)
summary(final_plm, test="Chisq")
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + parties + pollib, family = poisson,
##     data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.361  -1.041  -0.315   0.615   1.754
##
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2080     0.4457    0.47    0.641
## oligarchy     0.0915     0.0226    4.05    5e-05 ***
## parties       0.0224     0.0091    2.46    0.014 *
## pollib1      -0.4954     0.4757   -1.04    0.298
## pollib2      -1.1121     0.4595   -2.42    0.016 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 65.945  on 35  degrees of freedom
## Residual deviance: 32.822  on 31  degrees of freedom
## AIC: 107.6
##
## Number of Fisher Scoring iterations: 5
```

In comparison with a) all the same variables are significant.

c)

```
m_o <- mean(coups_df$oligarchy);
m_p <- mean(coups_df$parties)
```

```
newdata <- data.frame(pollib=c("0", "1", "2"), oligarchy=c(m_o, m_o, m_o), parties=c(m_p, m_p,
predict(final_plm, newdata, type="response")
```

```
##     1     2     3
## 2.908 1.772 0.956
```

Our model predicts that there will be roughly 3 successful coups for pollib=0, roughly 2 successful coups for pollib=1, and 1 successful coup for pollib=2.