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 a 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
```

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*.

```r
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
```

```r
summary(tree_type_d_lm)
```

```
##
## Call:
## lm(formula = volume ~ height + type * diameter, data = tree_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.350 -2.194 -0.141  1.701  8.176
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -63.873      5.539  -11.53  3.5e-16 ***
## height              0.434      0.079    5.49  1.1e-06 ***
## typeoak            -4.963      5.149   -0.96     0.34
## diameter            4.608      0.207   22.26  < 2e-16 ***
## typeoak:diameter    0.259      0.359    0.72     0.47
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.26 on 54 degrees of freedom
## Multiple R-squared:  0.951,  Adjusted R-squared:  0.948
## F-statistic:  264 on 4 and 54 DF,  p-value: <2e-16
```

```r
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
```

```r
summary(tree_type_h_lm)
```

```
##
## Call:
## lm(formula = volume ~ diameter + type * height, data = tree_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.230 -2.113 -0.161  1.801  8.165
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -57.551      7.111   -8.09    7e-11 ***
## diameter          4.779      0.173   27.55   <2e-16 ***
## typeoak         -17.471     11.826   -1.48   0.1454
## height            0.321      0.102    3.14   0.0027 **
## typeoak:height    0.212      0.154    1.37   0.1761
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.22 on 54 degrees of freedom
## Multiple R-squared:  0.953,  Adjusted R-squared:  0.949
## F-statistic:  271 on 4 and 54 DF,  p-value: <2e-16
```

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*. Both models suggest that all three explanatory variables have a significant effect individually.

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).

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

```
## Analysis of Variance Table
##
## Response: volume
##           Df Sum Sq Mean Sq F value  Pr(>F)
## diameter   1  10827   10827 1029.51 < 2e-16 ***
## height     1    346     346   32.92 4.3e-07 ***
## type       1     23      23    2.21    0.14
## Residuals 55    578      11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(tree_add_all_lm)
```

```
##
## Call:
## lm(formula = volume ~ diameter + height + type, data = tree_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.186 -2.140 -0.087  1.721  7.701
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -63.7814     5.5129  -11.57  2.3e-16 ***
## diameter      4.6981     0.1645   28.56  < 2e-16 ***
## height        0.4172     0.0752    5.55  8.4e-07 ***
## typeoak      -1.3046     0.8779   -1.49     0.14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.24 on 55 degrees of freedom
## Multiple R-squared:  0.951,  Adjusted R-squared:  0.948
## F-statistic:  355 on 3 and 55 DF,  p-value: <2e-16
```

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

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

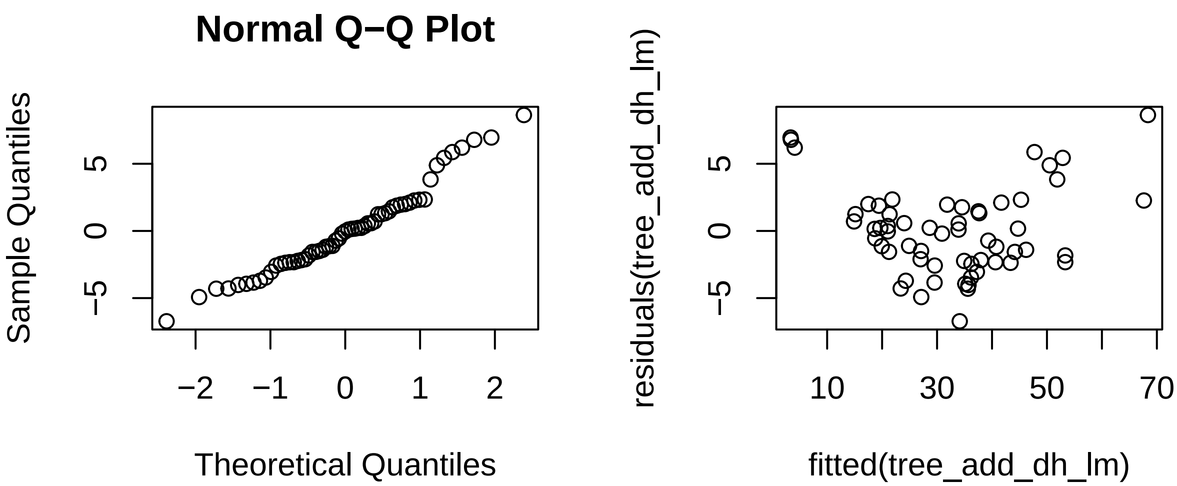
```
## Analysis of Variance Table
##
## Response: volume
##           Df Sum Sq Mean Sq F value  Pr(>F)
## diameter   1  10827   10827  1007.8 < 2e-16 ***
## height     1    346     346    32.2 5.1e-07 ***
## Residuals 56    602      11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1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

```r
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:

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

```r
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.

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.

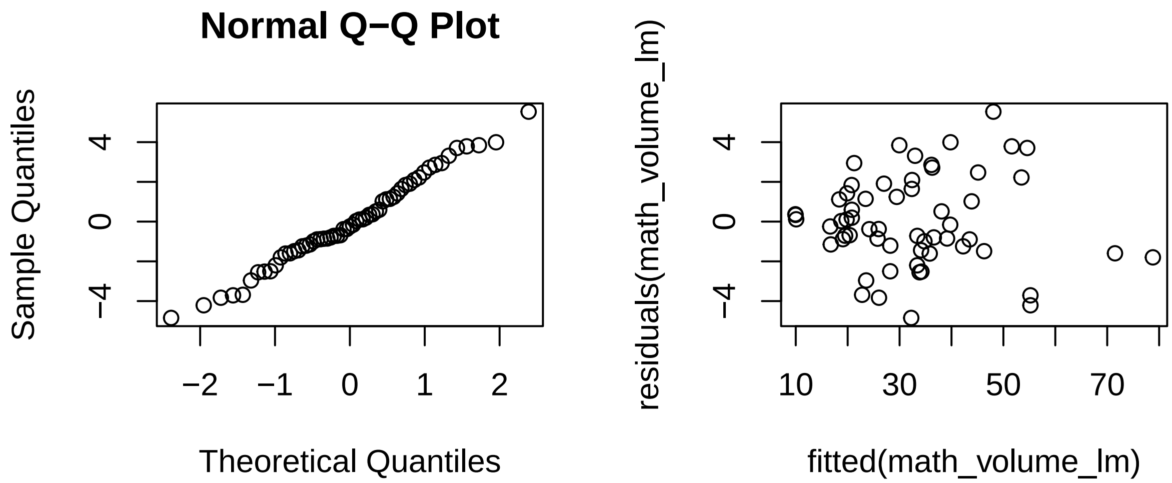
```
tree_df$math_volume <- tree_df$height * tree_df$diameter^2
math_volume_lm <- lm(volume~math_volume, data=tree_df)
anova(math_volume_lm)
```

```
## 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.

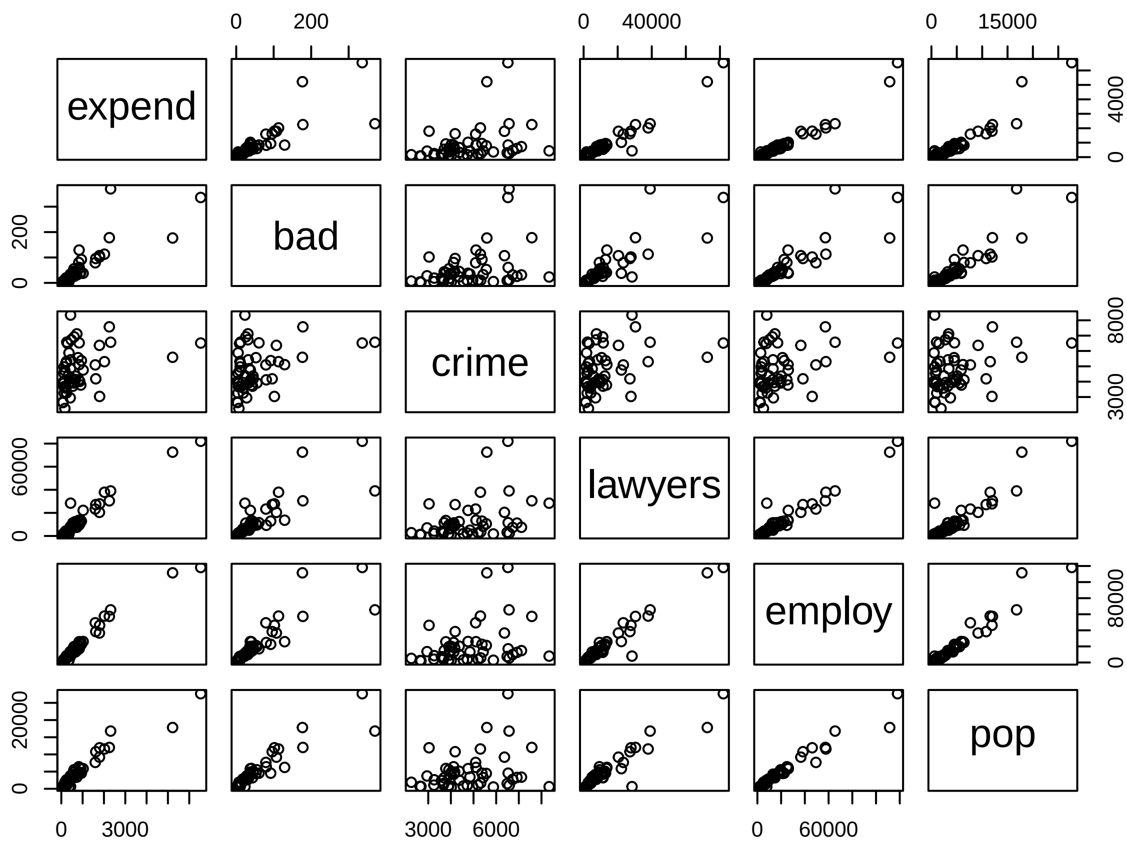


These plots are acceptable, meaning we can accept the model assumptions.

Exercise 2

a)

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

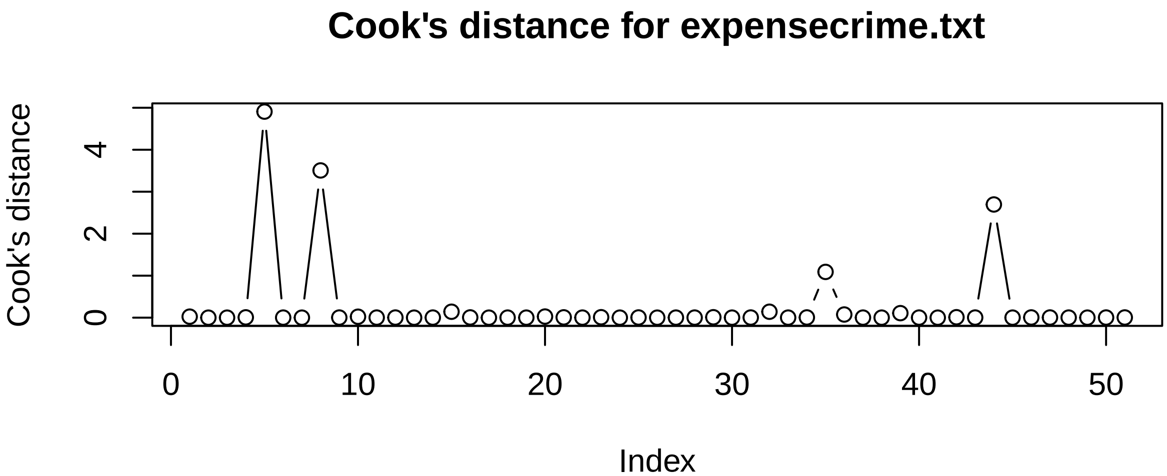


We see that *expend*, our response variable, appears to have a positive correlation with all the explanatory variables except for *crime*. There appear to be several outliers at the high end of the data which could skew the model. We can also see that collinearity exists between the explanatory variables *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
```

«««« IS VIF NECCESSARY? »»»»

To resolve the problem of collinearity, we can iteratively remove variables based on their VIF-values as follows:

Full model:

```
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
```

Remove *employ*:

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

```
##     bad   crime lawyers     pop 
##    7.16    1.34   12.40   20.72
```

Remove *pop*:

```
vif_lm = lm(expend~bad+crime+lawyers, data=crime_df_upd)
vif(vif_lm)
```

```
##     bad   crime lawyers 
##    3.99    1.13    3.78
```

```
summary(vif_lm)
```

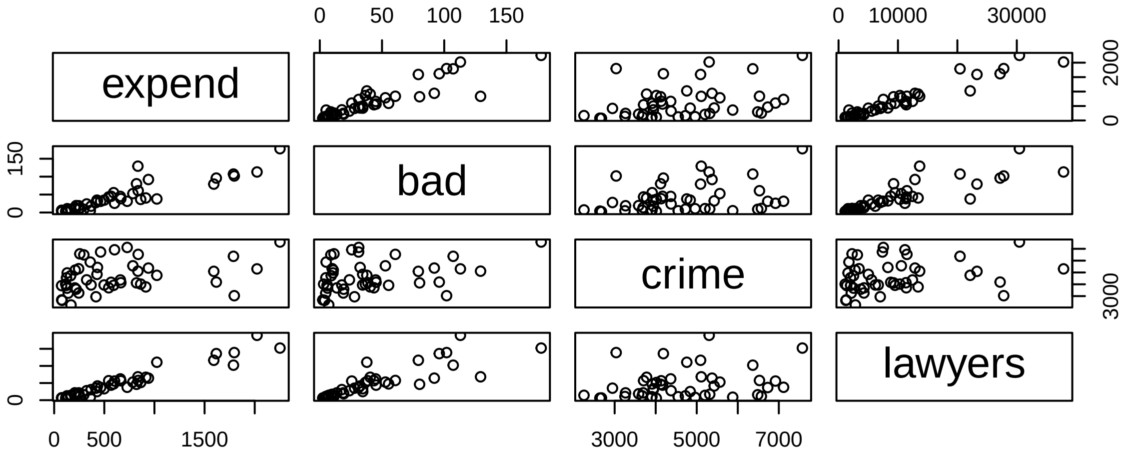
```
## 
## Call:
## lm(formula = expend ~ bad + crime + lawyers, data = crime_df_upd)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -328.4  -42.4  -14.2   33.8  355.3 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -113.5051    66.5587   -1.71  0.09535 .  
## bad            3.7457     0.8845    4.23  0.00012 ***
## crime          0.0333     0.0145    2.30  0.02655 *  
## lawyers        0.0456     0.0039   11.68  6.3e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 118 on 43 degrees of freedom
## Multiple R-squared:  0.957,  Adjusted R-squared:  0.954 
## F-statistic:  322 on 3 and 43 DF,  p-value: <2e-16
```

In the resulting model, all the explanatory variables are significant.

«««« SHOULD WE SHOW PLOT AGAIN?? »»»»

Therefore, after removing the influence points and collinear explanatory variables, the adjusted scatter plot appears as follows. We will work with this adjusted data for the remainder of this question.

```
pairs(crime_df_upd[, c(response, "bad", "crime", "lawyers")])
```



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:

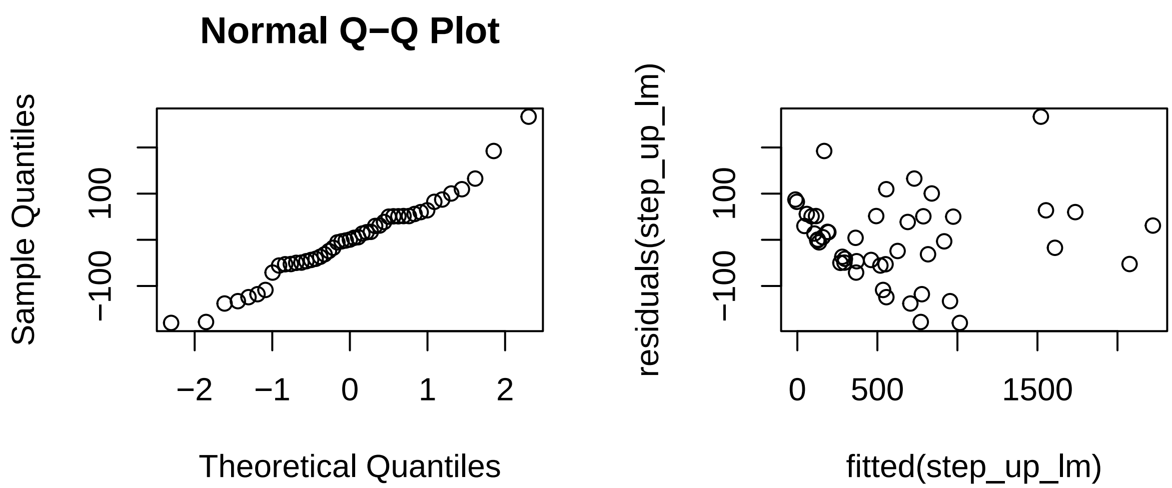
```
step_up_lm <- lm(expend~employ+crime+pop, data=crime_df_upd)
summary(step_up_lm)
```

```
##
## 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 $R^2 = 0.974$.

We see that the step-up method naturally removes collinearity and produced a better model than was arrived upon using VIF in (a), which had an R-squared value of 0.957.

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 can improve this interval (make it more narrow) by considering the **confidence interval**, which does not take into account the error.

```
predict(step_up_lm, new_data, interval="confidence", level=0.95)
```

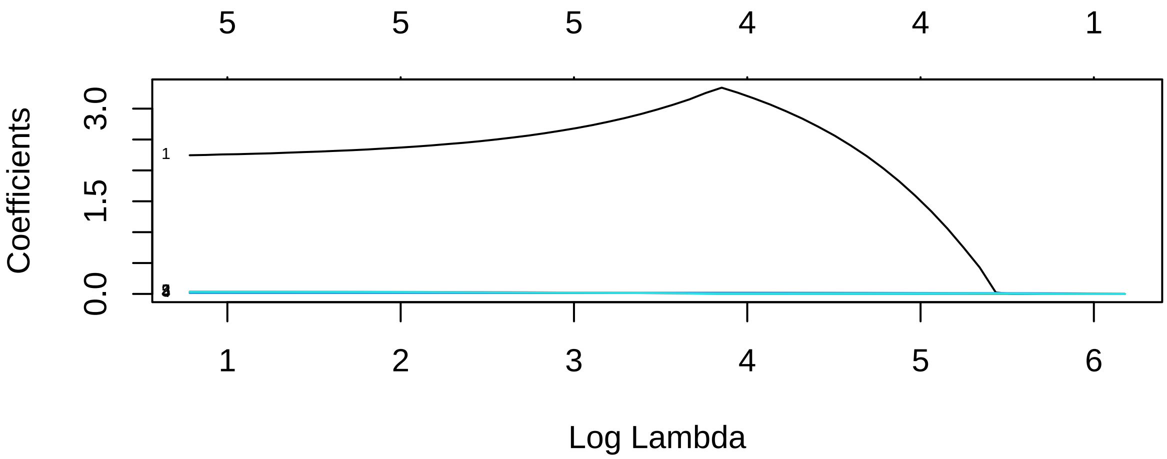
```
##   fit lwr upr
## 1 486 352 619
```

d)

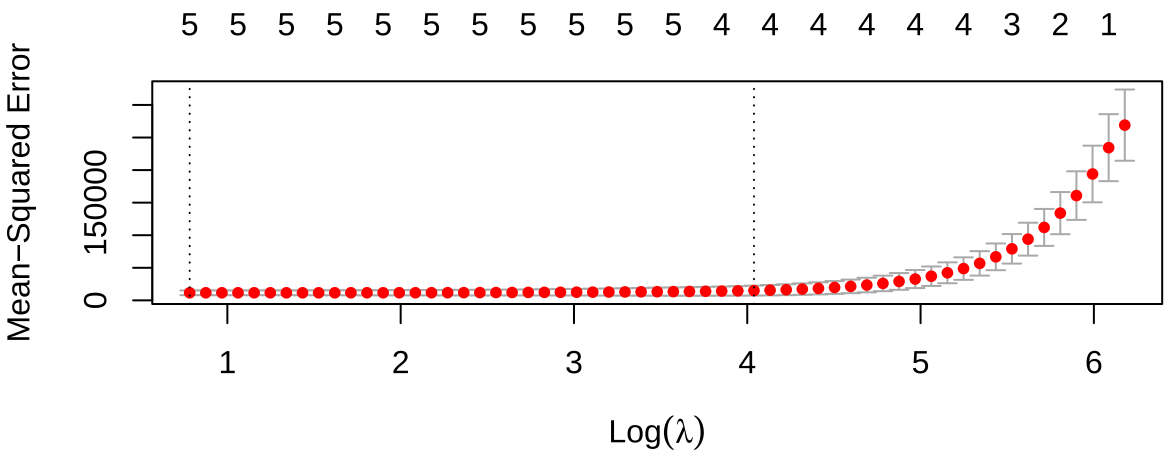
We can apply the lasso method as follows:

```
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")  # have a look at the lasso path
```




```
plot(cv.lasso) # the best lambda by cross-validation
```




```
(lambda.min <- cv.lasso$lambda.min)
```

```
## [1] 2.19
```

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

```
## [1] 56.8
```

```
# https://glmnet.stanford.edu/articles/glmnet.html#assessing-models-on-test-data-1
assess.glmnet(lasso.mod, newx = x.test, newy = y.test, s=cv.lasso$lambda.1se)
```

```
## $mse
##    s1
## 12518
## attr(,"measure")
## [1] "Mean-Squared Error"
##
## $mae
##   s1
## 79.4
## attr(,"measure")
## [1] "Mean Absolute Error"
```

Looking at lambda min

```
coef(lasso.mod, s=cv.lasso$lambda.min) # beta's for the best lambda
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                     s1
## (Intercept) -176.0429
## bad            2.2450
## crime          0.0371
## lawyers        0.0181
## employ         0.0150
## pop            0.0343
```

```
y.pred <- predict(lasso.mod, s=lambda.min, newx=x.test) # predict for test
mse.lasso <- mean((y.test - y.pred)^2); mse.lasso # mse for the predicted test rows
```

```
## [1] 12690
```

Looking at lambda 1se (the one we should use I think?)

```
coef(lasso.mod, s=cv.lasso$lambda.1se) # beta's for lambda.1se
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                   s1
## (Intercept) 62.91667
## bad          3.16571
## crime        .
## lawyers      0.02213
## employ       0.01356
## pop          0.00352
```

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

```
## [1] 12518
```

Compare to step-up model in (b).

```
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 # mse for the predicted test rows
```

```
## [1] 8231
```

Step-up model is better? Is this the right way to compare?

```
cv.lasso
```

```
##
## Call:  cv.glmnet(x = x.train, y = y.train, type.measure = "mse", alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure   SE Nonzero
```

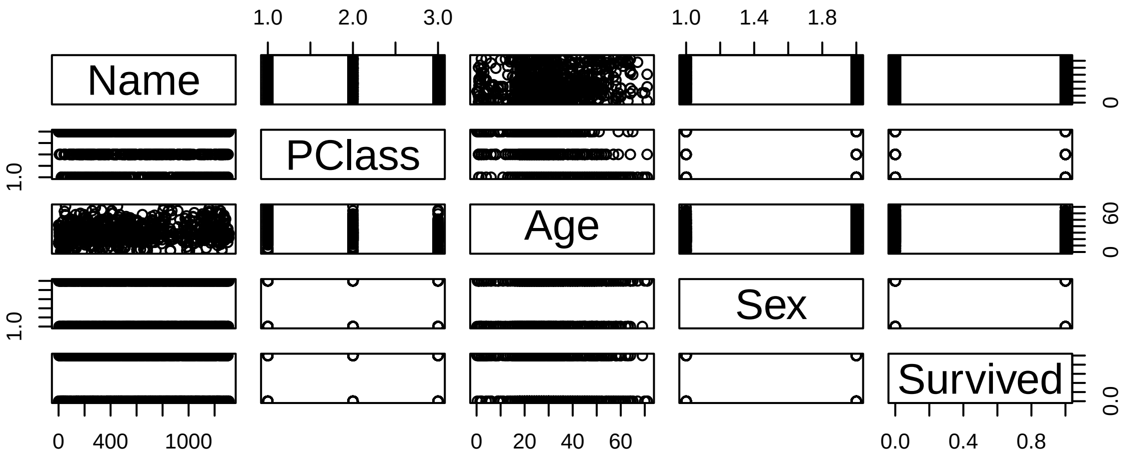
```
## min     2.2     59    11608 3624        5
## 1se    56.8     24    14981 7642        4
```

Exercise 3

```r
head(titanic_df)
```

```
##                                            Name PClass   Age    Sex Survived
## 1                  Allen, Miss Elisabeth Walton    1st 29.00 female        1
## 2                   Allison, Miss Helen Loraine    1st  2.00 female        0
## 3           Allison, Mr Hudson Joshua Creighton    1st 30.00   male        0
## 4 Allison, Mrs Hudson JC (Bessie Waldo Daniels)    1st 25.00 female        0
## 5                 Allison, Master Hudson Trevor    1st  0.92   male        1
## 6                            Anderson, Mr Harry    1st 47.00   male        1
```

```r
plot(titanic_df)
```



a)

```r
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
```

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

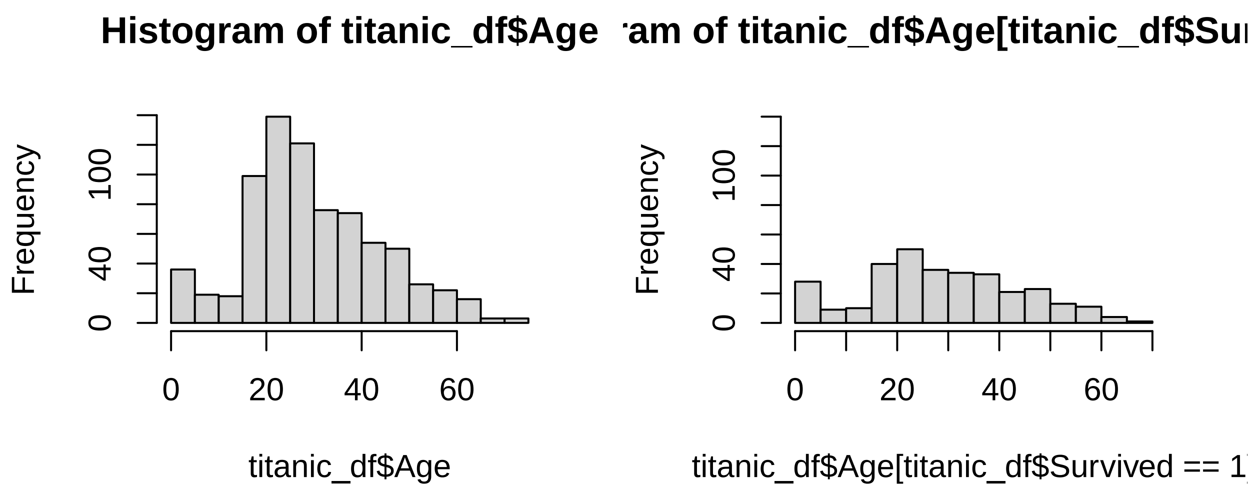
```
##       Sex
## PClass female male
```

```
##     1st     143  179
##     2nd     107  173
##     3rd     212  499
```

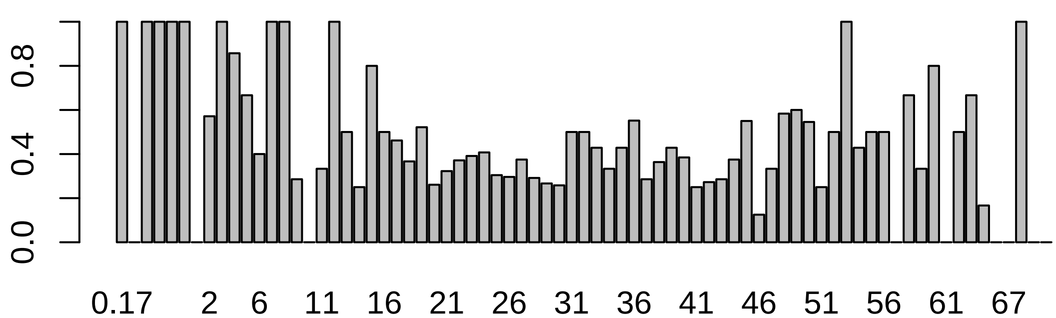
```
tot_comb.surv <- xtabs(Survived~PClass+Sex, data=titanic_df)
round(tot_comb.surv/tot_comb, 2)
```

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

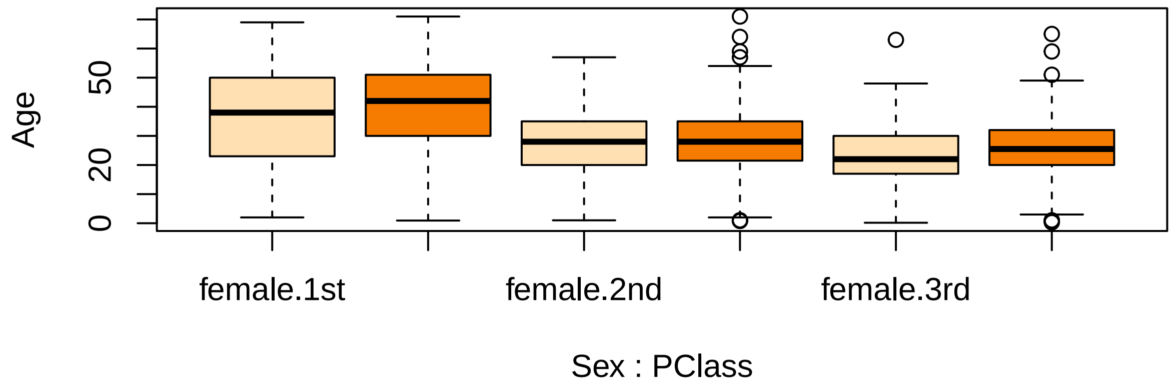
```
par(mfrow=c(1, 2))
hist(titanic_df$Age)
hist(titanic_df$Age[titanic_df$Survived == 1], ylim =c(0, 140))
```




```
tot_age <- xtabs(~Age, data=titanic_df)
barplot(xtabs(Survived~Age, data=titanic_df)/tot_age)
```




```
boxplot(Age ~ Sex + PClass, data=titanic_df, col = c("#FFE0B2", "#F57C00"))
```




```r
titanic_df_upd <- na.omit(titanic_df)
titanic_df_upd$PClass <- as.factor(titanic_df_upd$PClass)
titanic_df_upd$Sex <- as.factor(titanic_df_upd$Sex)
# head(titanic_df_upd)

base_lm <- glm(Survived ~ Age+PClass+Sex, data = titanic_df_upd, family = binomial)
```

```r
summary(base_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
```

```r
exp(coef(base_lm))
```

```
## (Intercept)         Age   PClass2nd   PClass3rd     Sexmale
##     42.9339      0.9616      0.2747      0.0803      0.0720
```

TODO: add discussion of odds from the paper

b)

```r
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
```

```r
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
```

Therefore we decided to keep following model as Age:Sex intersection is significant.

```r
final_lm <- glm(Survived ~ PClass+Age*Sex, data = titanic_df_upd, family = binomial)
anova(final_lm, 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
## PClass   2     78.0       753        948  < 2e-16 ***
## Age      1     37.6       752        910  8.6e-10 ***
## Sex      1    214.8       751        695  < 2e-16 ***
## Age:Sex  1     28.1       750        667  1.2e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

```
##      1      2      3      4      5      6
## 0.9474 0.1450 0.7937 0.0350 0.5590 0.0118
```

For "female" all the probs > 0.5 and for the "male" probs are < 0.5.

c) Use confusion matrix, log likelihood as quality measures

d)

```r
table(titanic_df_upd$PClass, titanic_df_upd$Sex)
```

```
##
##       female male
##   1st    101  125
##   2nd     85  127
##   3rd    102  216
```

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

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  titanic_df_upd$Survived and titanic_df_upd$Sex
## X-squared = 219, df = 1, p-value <2e-16
```

For 2x2 tables we can obtain exact p-value using the Fisher test.

```r
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
```

```
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
```

We reject null hypothesis, meaning that rows and cols are actually dependent: Sex and PClass have influence on Survived variable.

e)

TODO: comparison btw c) and d) ?

contingency table tells us only about the presence of effect and doesn't provide some quantitative characteristics

Exercise 4

```
head(coups_df)
```

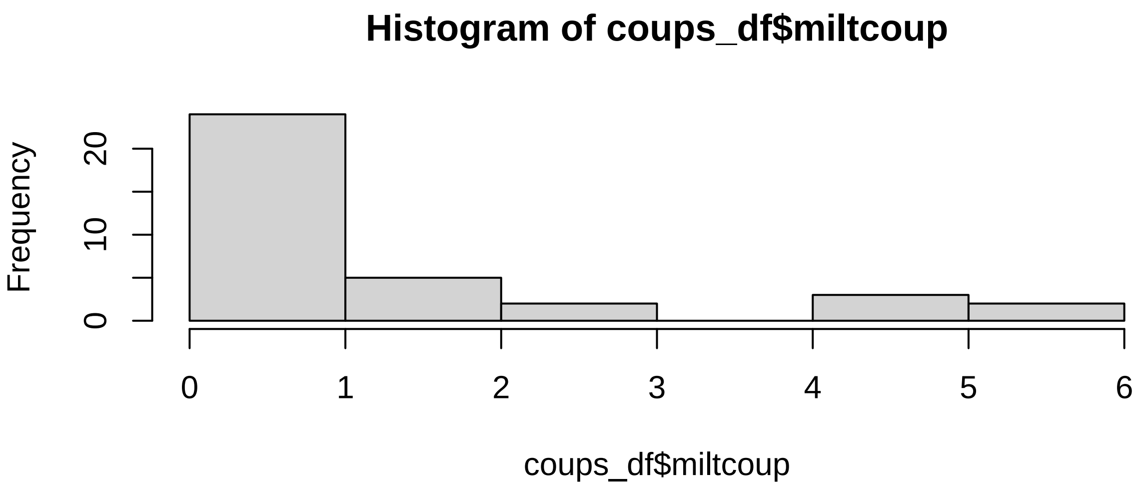
```
##           miltcoup oligarchy pollib parties pctvote   popn size numelec
## Benin            5         7      1      34    45.7  4.600  113       8
## Burkina          6        13      2      62    17.5  8.800  274       5
## Burundi          2        13      2      10    34.4  5.300   28       3
## Cameroon         0         0      2      34    30.3 11.600  475      14
## Capeverde        1         0      2       5    30.5  0.361    4       2
## CAR              3        14      2      14    16.2  3.000  623       6
##           numregim
## Benin            3
## Burkina          3
## Burundi          3
## Cameroon         3
## Capeverde        1
## CAR              4
```

```
# coups_df$pollib <- as.factor(coups_df$pollib)
# coups_df$numregim <- as.factor(coups_df$numregim)
summary(coups_df)
```

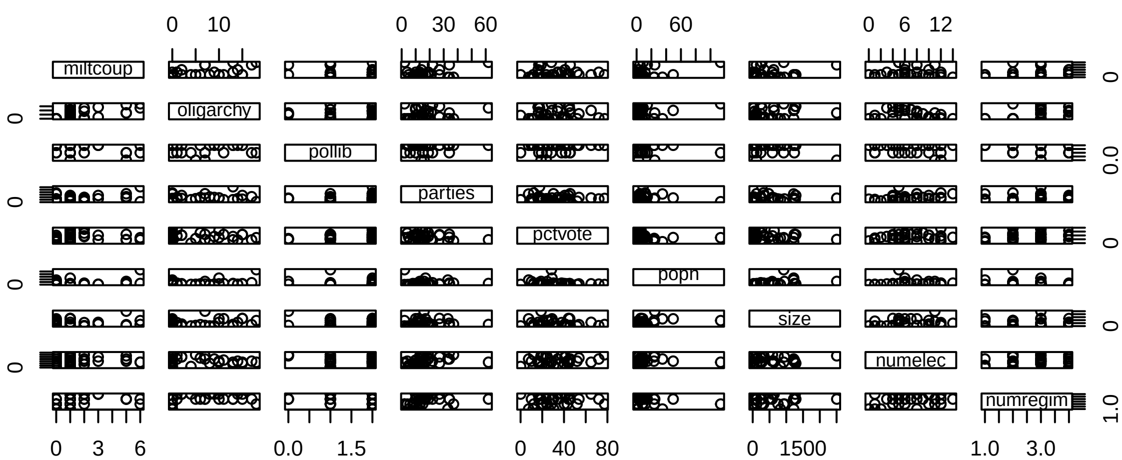
```
##     miltcoup       oligarchy         pollib         parties         pctvote
```

```
##  Min.   :0.00   Min.   : 0.00   Min.   :0.00   Min.   : 2.0   Min.   : 0.0
##  1st Qu.:0.00   1st Qu.: 0.00   1st Qu.:1.00   1st Qu.:10.0   1st Qu.:18.6
##  Median :1.00   Median : 2.00   Median :2.00   Median :14.0   Median :29.6
##  Mean   :1.58   Mean   : 5.22   Mean   :1.64   Mean   :17.1   Mean   :32.1
##  3rd Qu.:2.00   3rd Qu.:10.00   3rd Qu.:2.00   3rd Qu.:19.5   3rd Qu.:43.3
##  Max.   :6.00   Max.   :18.00   Max.   :2.00   Max.   :62.0   Max.   :77.4
##       popn            size         numelec        numregim
##  Min.   :  0.1   Min.   :   0   Min.   : 0.00   Min.   :1.00
##  1st Qu.:  1.7   1st Qu.:  34   1st Qu.: 4.00   1st Qu.:2.00
##  Median :  5.6   Median : 271   Median : 6.00   Median :3.00
##  Mean   : 11.6   Mean   : 485   Mean   : 6.72   Mean   :2.75
##  3rd Qu.: 11.4   3rd Qu.: 771   3rd Qu.:10.00   3rd Qu.:3.25
##  Max.   :113.8   Max.   :2506   Max.   :14.00   Max.   :4.00
```

```
hist(coups_df$miltcoup)
```




```
plot(coups_df)
```



a)

```
poison_glm <- glm(miltcoup ~ oligarchy + pollib + parties + pctvote + popn + size + numelec + n
summary(poison_glm)
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + pctvote +
##     popn + size + numelec + numregim, family = poisson, data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.344  -0.954  -0.259   0.391   1.695
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.510269   0.905330   -0.56   0.5730
## oligarchy    0.073081   0.034596    2.11   0.0346 *
## pollib      -0.712978   0.272563   -2.62   0.0089 **
## parties      0.030774   0.011187    2.75   0.0059 **
## pctvote      0.013872   0.009753    1.42   0.1549
## popn         0.009343   0.006595    1.42   0.1566
## size        -0.000190   0.000248   -0.76   0.4445
## numelec     -0.016078   0.065484   -0.25   0.8060
## numregim     0.191735   0.229289    0.84   0.4030
## ---
## 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: 28.668  on 27  degrees of freedom
## AIC: 111.5
##
## Number of Fisher Scoring iterations: 6
```

Through summary(drop1)we can find the variables that are significant in predicting number of successful military coups: oligarchy, pollib, parties.

b)

```
summary(glm(miltcoup ~ oligarchy + pollib + parties + pctvote + popn + size + numelec, data = c
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + pctvote +
##     popn + size + numelec, family = poisson, data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.353  -0.965  -0.195   0.483   1.617
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -0.114181   0.756715   -0.15   0.8801
## oligarchy    0.086007   0.030774    2.79   0.0052 **
## pollib      -0.689010   0.270392   -2.55   0.0108 *
## parties      0.029183   0.011006    2.65   0.0080 **
## pctvote      0.014150   0.009753    1.45   0.1468
## popn         0.006272   0.005440    1.15   0.2490
## size        -0.000195   0.000247   -0.79   0.4297
## numelec      0.000168   0.062185    0.00   0.9978
## ---
## 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: 29.363  on 28  degrees of freedom
## AIC: 110.2
##
## Number of Fisher Scoring iterations: 5
```

```
summary(glm(miltcoup ~ oligarchy + pollib + parties + pctvote + popn + size, data = coups_df, 
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + pctvote +
##     popn + size, family = poisson, data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.352  -0.965  -0.195   0.483   1.618
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.112687   0.516303   -0.22  0.82723
## oligarchy    0.085962   0.025910    3.32  0.00091 ***
## pollib      -0.689403   0.227857   -3.03  0.00248 **
## parties      0.029194   0.010195    2.86  0.00419 **
## pctvote      0.014159   0.009198    1.54  0.12372
## popn         0.006274   0.005399    1.16  0.24527
## size        -0.000195   0.000242   -0.80  0.42138
## ---
## 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: 29.363  on 29  degrees of freedom
## AIC: 108.2
##
```

```
## Number of Fisher Scoring iterations: 5
```

```r
summary(glm(miltcoup ~ oligarchy + pollib + parties + pctvote + popn, data = coups_df), test="(
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + pctvote +
##     popn, data = coups_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0424  -0.8843  -0.0798   1.1548   2.0044
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.0174     0.7904    1.29   0.2079
## oligarchy     0.1277     0.0393    3.25   0.0028 **
## pollib       -1.0098     0.3811   -2.65   0.0127 *
## parties       0.0504     0.0193    2.62   0.0138 *
## pctvote       0.0152     0.0126    1.20   0.2393
## popn          0.0178     0.0115    1.55   0.1325
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.59)
##
##     Null deviance: 108.750  on 35  degrees of freedom
## Residual deviance:  47.787  on 30  degrees of freedom
## AIC: 126.4
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(glm(miltcoup ~ oligarchy + pollib + parties + popn, data = coups_df), test="Chisq")
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + popn,
##     data = coups_df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.928  -0.800  -0.220   0.987   2.208
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.4201     0.7209    1.97   0.0578 .
## oligarchy     0.1343     0.0392    3.43   0.0017 **
## pollib       -0.9083     0.3743   -2.43   0.0212 *
## parties       0.0454     0.0190    2.40   0.0228 *
```

```
## popn          0.0150     0.0113    1.33   0.1947
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.62)
##
##     Null deviance: 108.750  on 35  degrees of freedom
## Residual deviance:  50.083  on 31  degrees of freedom
## AIC: 126
##
## Number of Fisher Scoring iterations: 2
```

```
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
## -1.358  -1.042  -0.286   0.628   1.752
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.25138    0.37269    0.67    0.500
## oligarchy    0.09262    0.02178    4.25  2.1e-05 ***
## parties      0.02206    0.00896    2.46    0.014 *
## pollib      -0.57410    0.20438   -2.81    0.005 **
## ---
## 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: 32.856  on 32  degrees of freedom
## AIC: 105.7
##
## Number of Fisher Scoring iterations: 5
```

After step-down approach there are oligarchy, parties and pollib left. (For treating vars as factors only oligarchy left) In comparison with a) all the same factors are significant.

c)

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

```
mean(coups_df$oligarchy); mean(coups_df$parties)
```

```
## [1] 5.22
```

```
## [1] 17.1
```

```
newdata <- data.frame(pollib=c(0, 1, 2), oligarchy=c(5.22, 5.22, 5.22), parties=c(17.1, 17.1, 1
predict(final_plm, newdata, type="response")
```

```
##     1     2     3
## 3.041 1.713 0.965
```

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