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
```

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

```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
```

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*.

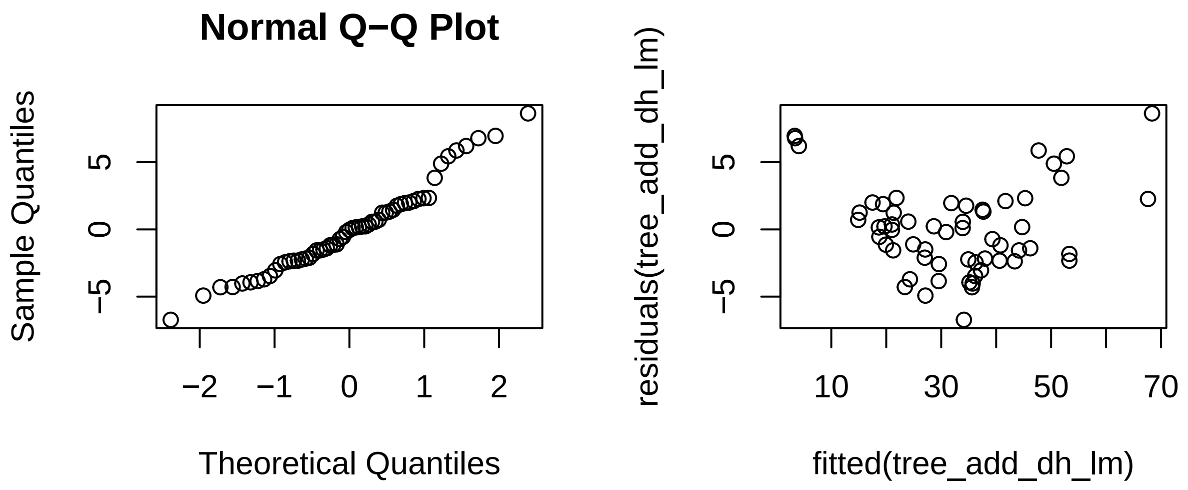
```
tree_add_dh_lm <- lm(volume~diameter+height, data=tree_df)
anova(tree_add_dh_lm)
```

```
## 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.

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.

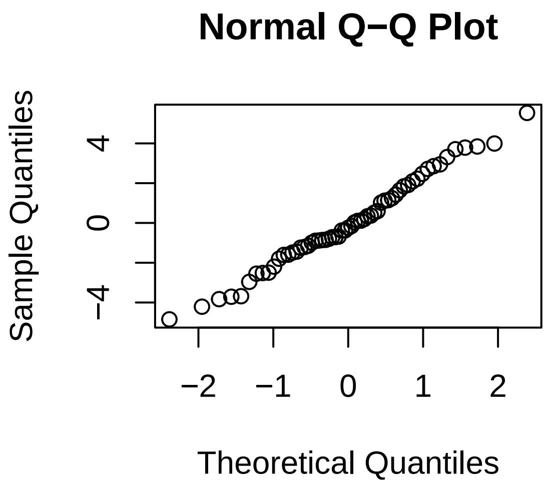
```
tree_df$math_volume <- tree_df$height * tree_df$diameter^2
math_volume_lm <- lm(volume~math_volume, data=tree_df)
anova(math_volume_lm)
```

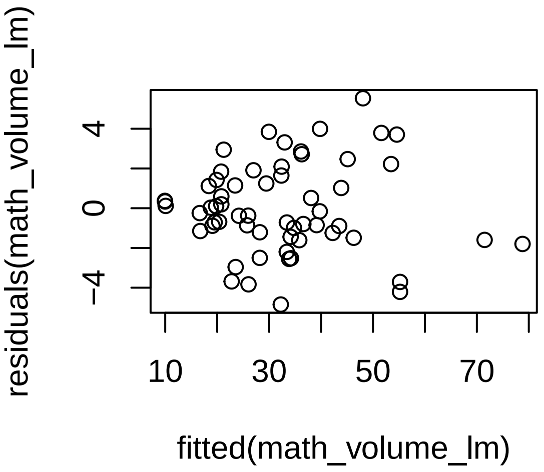
```
## Analysis of Variance Table
##
## Response: volume
##             Df Sum Sq Mean Sq F value Pr(>F)
## math_volume  1  11477   11477    2201 <2e-16 ***
## Residuals   57    297       5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
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.



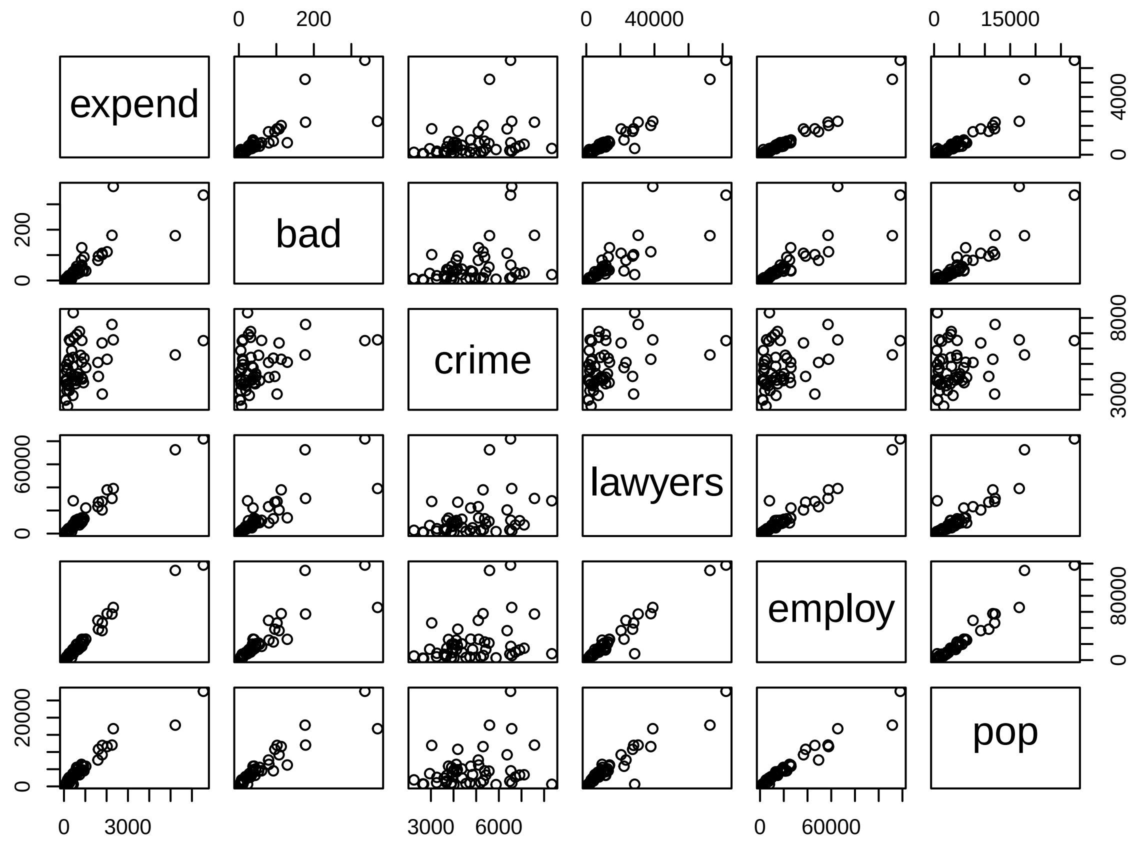


These plots are acceptable, meaning we can accept the model assumptions.

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*.

```r
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).

```r
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:

```r
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
```

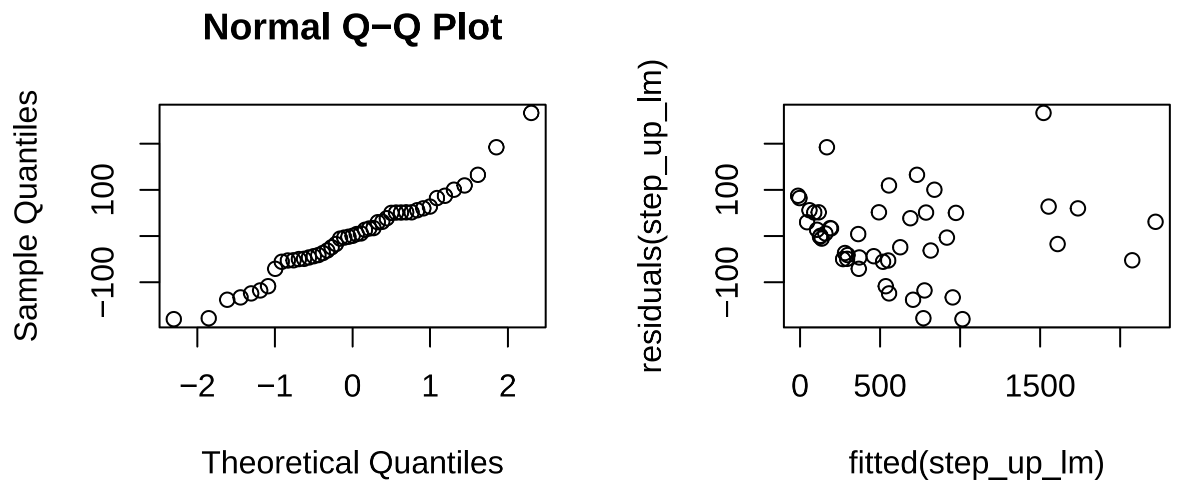
Final model: expend = -247 + 0.0209*employ + 0.0543*crime + 0.0714*pop ± error, with R^2 = 0.974. Using VIF we can

```
vif(step_up_lm)
```

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

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

d)

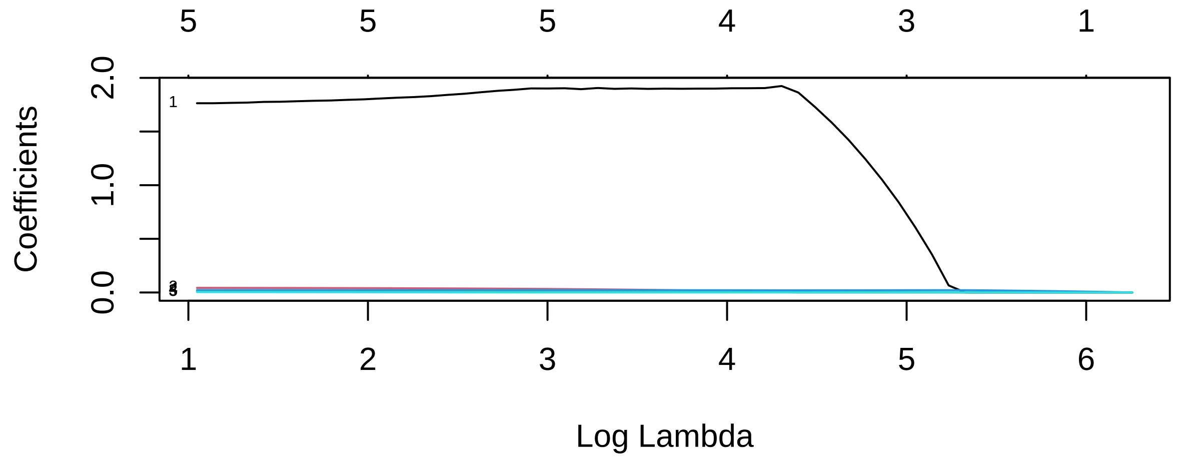
We can apply the LASSO method as follows:

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

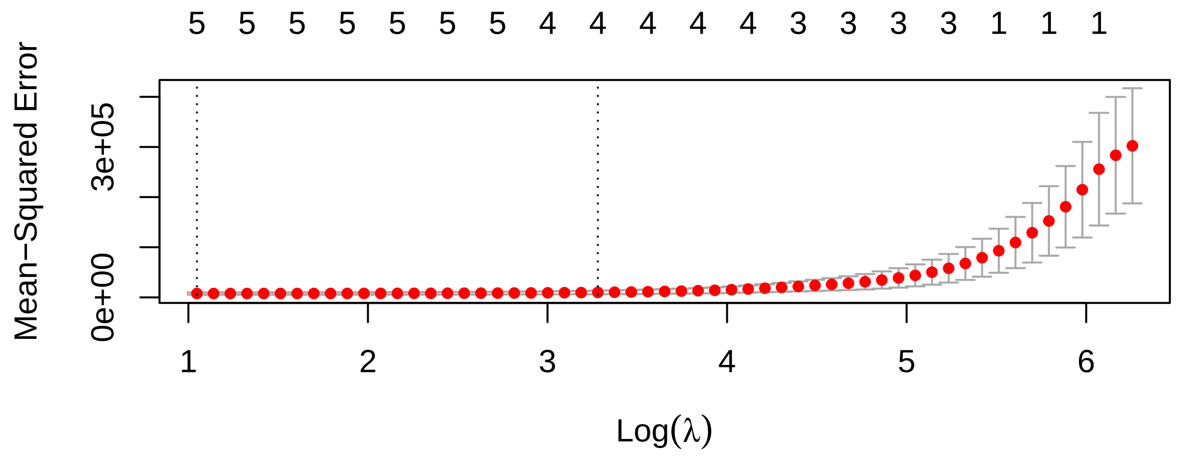
```
# train-test splitting
train <- (sample(1:nrow(x), 0.67*nrow(x))) # train by using 2/3 of the data
x.train <- x[train,]; y.train <- y[train]
x.test <- x[-train,]; y.test <- y[-train]

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



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



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

```
## [1] 26.6
```

```r
# 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
## 23960
## attr(,"measure")
## [1] "Mean-Squared Error"
##
## $mae
##  s1
## 104
## attr(,"measure")
## [1] "Mean Absolute Error"
```

Looking at lambda 1se

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

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                   s1
## (Intercept) -59.9643
## bad           1.9059
## crime         0.0290
## lawyers       0.0113
## employ        0.0219
## pop           .
```

```r
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] 23960
```

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

```r
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] 13322
```

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.

Exercise 3

a)

```
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
```

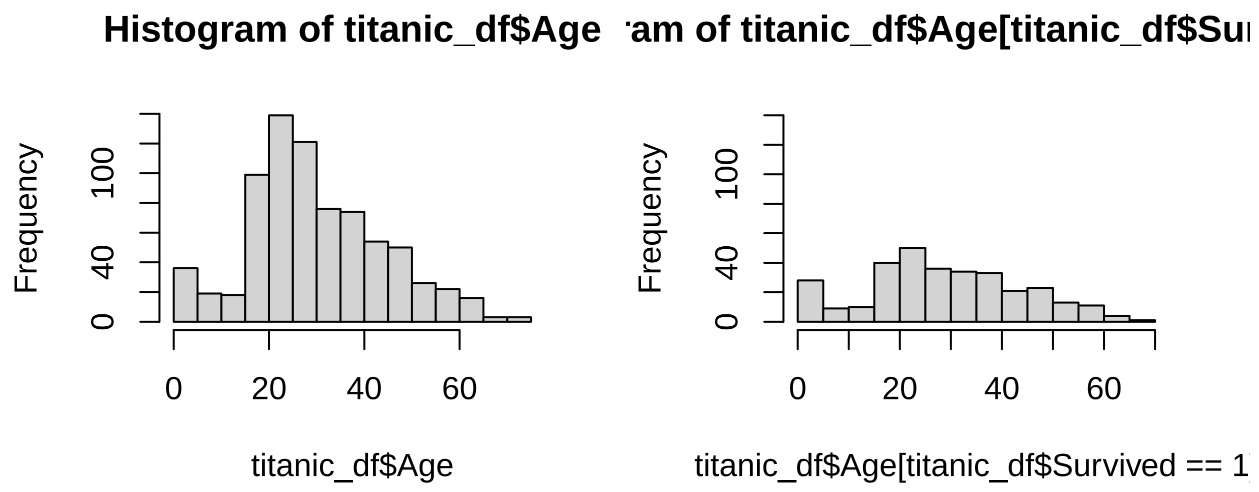
```
tot_comb <- xtabs(~PClass+Sex, data=titanic_df)
tot_comb
```

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

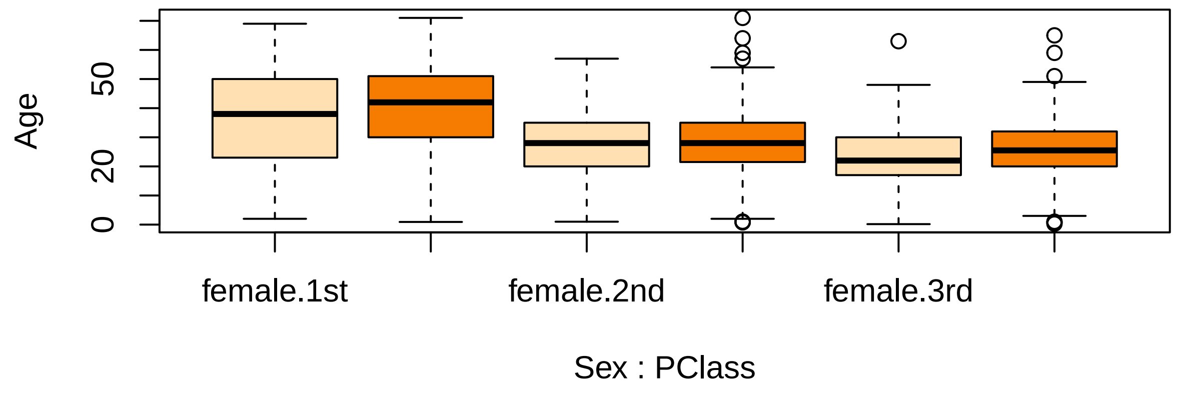
```
tot_comb.surv <- xtabs(Survived~PClass+Sex, data=titanic_df)
round(tot_comb.surv/tot_comb, 2)
```

```
##       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))
```



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



Removing rows with missing ages:

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

Fitting the logistic regression model.

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

```r
drop1(base_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
```

```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)

```
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
```

```
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 interaction is significant. PClass was significant by itself, so we include it in the final model.

««< INVESTIGATE VARIABLES USING anova(smaller model, bigger model) »»»

```
final_lm <- glm(Survived ~ PClass+Age*Sex, data = titanic_df_upd, family = binomial)
summary(final_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
```

«««< THIS MODEL MAY CHANGE »»>

```
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)

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.

```
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.

```
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.

e)

Logistic

A - Also tells you *how* and *how much* the response depends on the variable D - https://careerfoundry.com/en/blog/data-analytics/what-is-logistic-regression/

vs

Contingency

A - since only testing for effect, maybe more reliable outcome? D - Only tells you whether dependence exists… not the nature of it

vs

Fisher

A - exact p-value D - only works for 2x2

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

Exercise 4

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

a)

```
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
```

Through Poisson regression, and its summary, we can find that the variables that are significant in predicting 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:

```r
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)

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

```
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, 
predict(final_plm, newdata, type="response")
```

```
##     1     2     3
## 2.909 1.772 0.957
```

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.