EDDA Assignment2 - Group 22

Adwitiya Mandal, Oromia Sero, Priyakshi Goswami

2023-03-15

Exercise 1 a)

```
tree = read.table(file = "treeVolume.txt", header = TRUE)
tree$type = as.factor(tree$type)
```

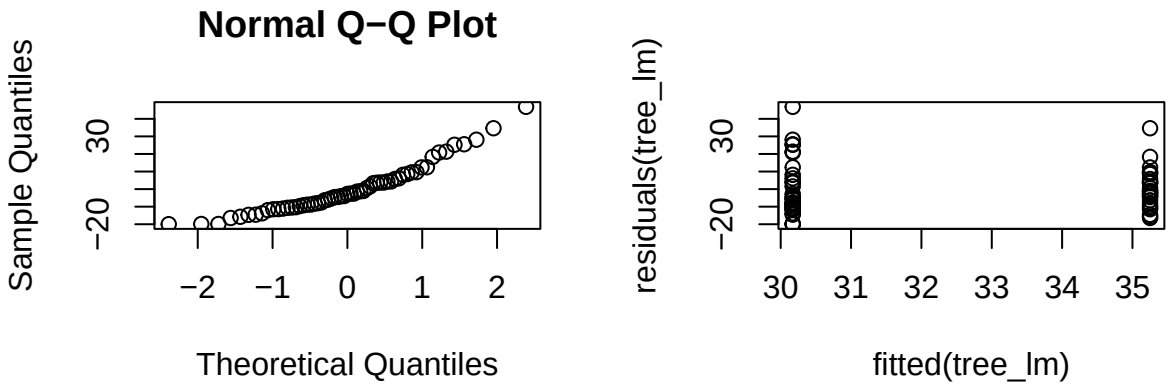
ANOVA: The null hypothesis is that the mean volume of type *beech* is equal to the mean volume of type *oak*.

```
tree_lm = lm(volume ~ type, data = tree)
anova(tree_lm)
```

```
## Analysis of Variance Table
##
## Response: volume
##           Df  Sum Sq Mean Sq F value Pr(>F)
## type       1   379.5  379.52  1.8984 0.1736
## Residuals 57 11394.8  199.91
```

The p-value (0.1736) is not significant. So H_0 cannot be rejected and we cannot say that tree type has any significant influence on volume.

Diagnostics for ANOVA



The residuals QQ-norm plot looks normal and the plot of the fitted model and residuals show no pattern. Hence, the assumptions of ANOVA hold.

```
summary(tree_lm)$coefficients
```

```
##              Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 30.170968   2.539425 11.88102 4.681462e-17
## typeoak      5.079032   3.686229  1.37784 1.736384e-01
```

We can estimate the volumes of the two tree types. For type:*beech*, $Volume = \mu + \alpha_1 = 30.171$ and for type:*oak*, $Volume = \mu + \alpha_2 = 30.171 + 5.079 = 35.25$.

t-test Yes, the t-test is related to the above test; because we only have one factor with two levels, we can do a two-sample t-test to see if there is a difference in mean tree volume among these tree types and end up with the same conclusion.

```
library(tidyr)
tree_table_2 <- tree[, c("type", "volume")]
t.test(tree_table_2[tree_table_2$type == "beech", ]$volume, tree_table_2[tree_table_2$type ==
    "oak", ]$volume, paired = FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  tree_table_2[tree_table_2$type == "beech", ]$volume and tree_table_2[tree_table_2$type == "oa
## t = -1.4051, df = 52.804, p-value = 0.1659
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -12.32992   2.17186
## sample estimates:
## mean of x mean of y
##  30.17097  35.25000
```

T-test also not rejected, the mean volume of the two types of trees is not significantly different. **b)**

```
tree_model1 <- lm(volume ~ diameter * type, data = tree)
summary(tree_model1)$coefficients
```

```
##                     Estimate Std. Error     t value     Pr(>|t|)
## (Intercept)      -36.9434591  3.1890267 -11.5845561 2.217267e-16
## diameter           5.0658564  0.2344302  21.6092285 1.506778e-28
## typeoak            2.8090341  6.1252469   0.4585993 6.483290e-01
## diameter:typeoak  -0.3250953  0.4241775  -0.7664134 4.467075e-01
```

We see that *diameter* has a significant influence on *volume*. But the interaction between *type* and *diameter* is not significant (p-value > 0.05).This means that there is no significant dependence between the two. So, there is no significant difference in the influence of diameter on volume for the two different tree types. They are similar.

```
tree_model2 <- lm(volume ~ height * type, data = tree)
summary(tree_model2)$coefficients
```

```
##                   Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)    -87.123614 27.0249998 -3.223816 2.129291e-03
## height           1.543350  0.3543888  4.354962 5.848926e-05
## typeoak         98.699466 42.4745519  2.323732 2.386271e-02
## height:typeoak  -1.230525  0.5586737 -2.202582 3.183357e-02
```

The above results suggests significant influence of height on volume as well as the interaction on the volume. This means the effect of height on the volume depends on the type of tree.

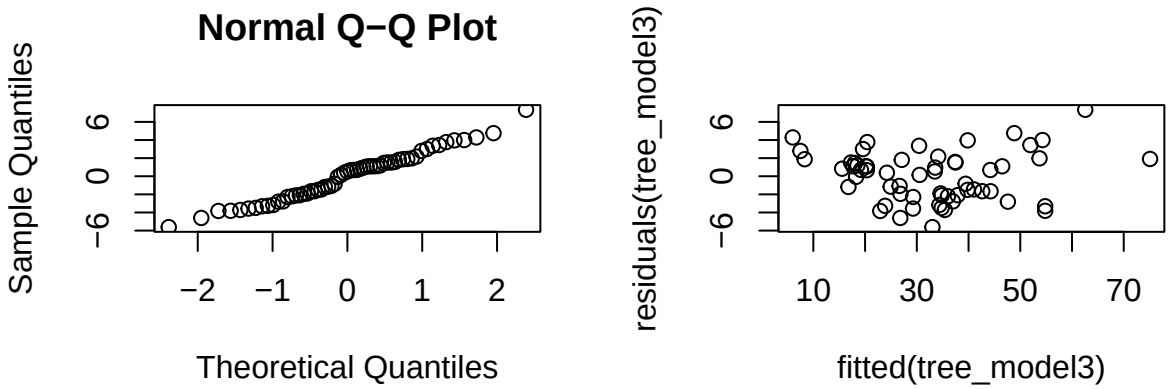
```
tree_model3 <- lm(volume ~ height * diameter, data = tree)
summary(tree_model3)
```

```
##
## Call:
## lm(formula = volume ~ height * diameter, data = tree)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.6159 -2.1025  0.5296  1.6942  7.3380
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     26.45042   19.76957   1.338  0.18642
## height          -0.79381    0.26630  -2.981  0.00427 **
## diameter        -1.86698    1.38049  -1.352  0.18178
## height:diameter  0.08713    0.01842   4.731  1.6e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.788 on 55 degrees of freedom
## Multiple R-squared:  0.9637, Adjusted R-squared:  0.9617
## F-statistic: 486.5 on 3 and 55 DF,  p-value: < 2.2e-16
```

The above model also suggests a very significant interaction between *height* and *diameter*. **c)** We also try including the type to the *tree_model*3 above. But we can see that now, the it is not significant enough and the R^2 value doesn't change and adjusted-R^2 slightly decreases. So we prefer *tree_model*3 without the type factor.

```
tree_model4 <- lm(volume ~ height * diameter + type, data = tree)
summary(tree_model4)$adj.r.squared
```

```
## [1] 0.9612065
```



Predicting Volume

```
avg_diameter = mean(tree$diameter)
avg_height = mean(tree$height)
volume_oak = predict(tree_model3, (data.frame(diameter = avg_diameter,
    height = avg_height, type = "oak")), type = "response")[[1]]
volume_beech = predict(tree_model3, (data.frame(diameter = avg_diameter,
    height = avg_height, type = "beech")), type = "response")[[1]]
volume_oak
```

```
## [1] 32.1868
```

```
volume_beech
```

```
## [1] 32.1868
```

d) We know that volume of a tree trunk can be approximately written as: $\pi * (diameter/2)^2 * height$. So, we propose a transformation $diameter \rightarrow diameter^2$.

```
tree$diametersq = tree$diameter * tree$diameter
final_model = lm(volume ~ diametersq * height, data = tree)
summary(final_model)
```

```
##
## Call:
## lm(formula = volume ~ diametersq * height, data = tree)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4173 -1.1104 -0.2242  1.4632  5.3837
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -3.894320   8.794942  -0.443     0.66
## diametersq        -0.002424   0.038491  -0.063     0.95
## height             0.050646   0.118361   0.428     0.67
## diametersq:height  0.002155   0.000511   4.217  9.3e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.303 on 55 degrees of freedom
## Multiple R-squared:  0.9752, Adjusted R-squared:  0.9739
## F-statistic: 721.7 on 3 and 55 DF,  p-value: < 2.2e-16
```

Clearly, this is a better model with a higher R^2 and adjusted-R^2 value. The interaction between diameter-square is significant too.

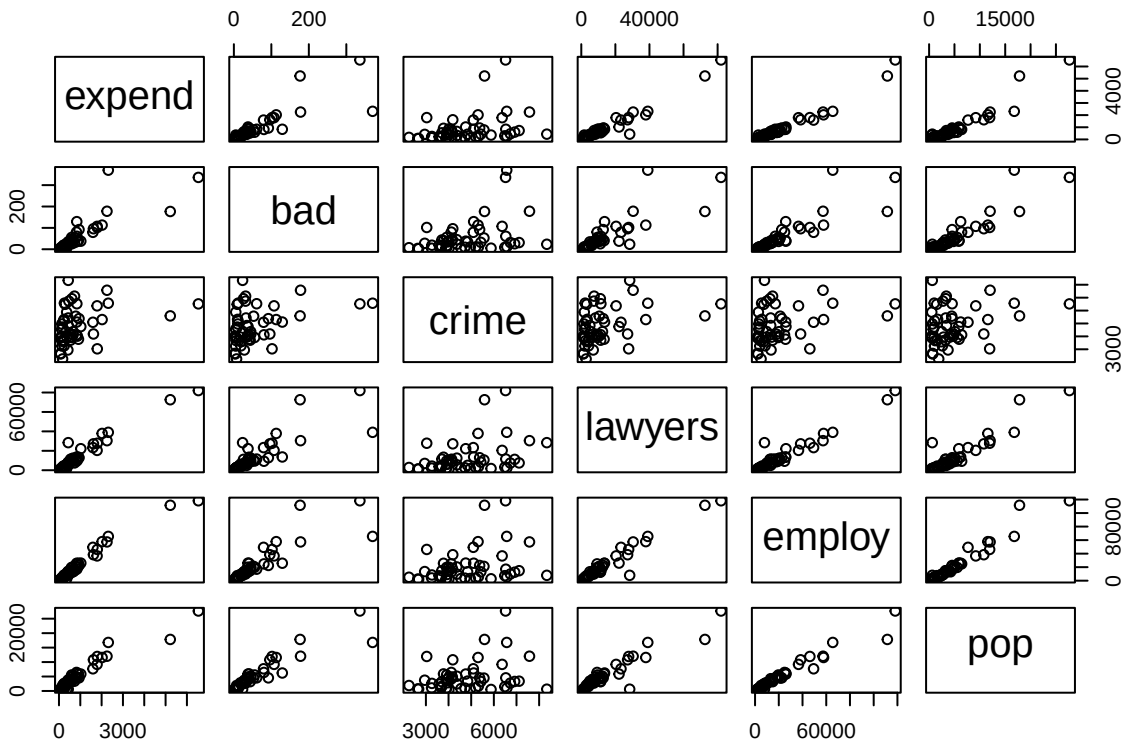
Exercise 2 a)

```
expense = read.table("expensescrime.txt", header = TRUE)
```

```
summary(expense)
```

```
##     state               expend            bad             crime
##  Length:51          Min.   :  74.0   Min.   :  2.40   Min.   :2253
##  Class :character   1st Qu.: 214.5   1st Qu.: 10.30   1st Qu.:3898
##  Mode  :character   Median : 463.0   Median : 31.30   Median :4549
##                     Mean   : 847.8   Mean   : 54.12   Mean   :4802
##                     3rd Qu.: 850.5   3rd Qu.: 58.00   3rd Qu.:5576
##                     Max.   :6539.0   Max.   :370.10   Max.   :8339
##     lawyers          employ            pop
##  Min.   : 1116   Min.   :  1969   Min.   :  490
##  1st Qu.: 2811   1st Qu.:  5304   1st Qu.: 1135
##  Median : 7535   Median : 13167   Median : 3296
##  Mean   :12892   Mean   : 20602   Mean   : 4773
##  3rd Qu.:13128   3rd Qu.: 23692   3rd Qu.: 5693
##  Max.   :82001   Max.   :118149   Max.   :27663
```

```
pairs(expense[, -1])
```




```
round(cor(expense[, -1]), 2)
```

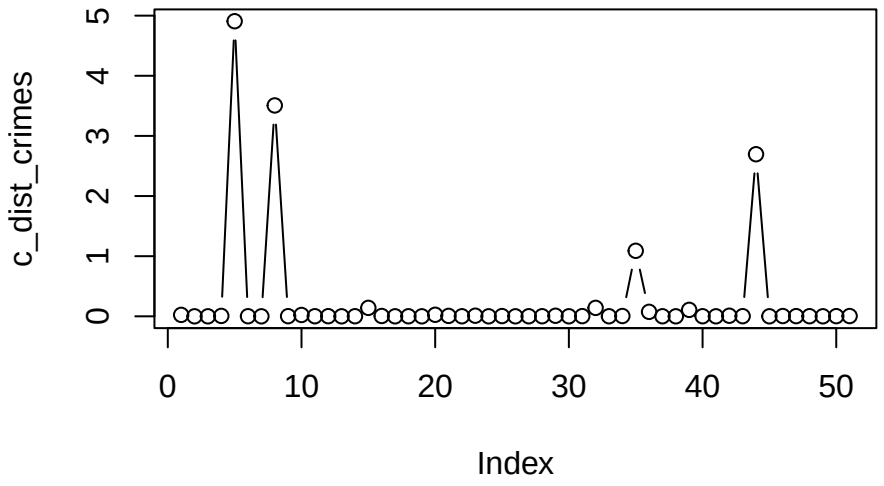
```
##         expend  bad crime lawyers employ  pop
## expend    1.00 0.83  0.33    0.97    0.98 0.95
## bad       0.83 1.00  0.37    0.83    0.87 0.92
```

```
## crime     0.33 0.37  1.00    0.38   0.31 0.28
## lawyers   0.97 0.83  0.38    1.00   0.97 0.93
## employ    0.98 0.87  0.31    0.97   1.00 0.97
## pop       0.95 0.92  0.28    0.93   0.97 1.00
```

The following pairs seems to have high collinearity: expend-lawyers(0.97), expend-employ(0.98), expend-pop(0.95), bad-pop(0.92), pop-lawyers(0.93),employ-lawyers(0.97), employ-pop(0.97).

Investigating Influence points: From the pairwise scatterplots, we can clearly see linear relations between many variables. To check influence points we can do cooks distance

```
c_dist_crimes = cooks.distance(lm(expend ~ bad + crime + lawyers +
    employ + pop, data = expense))
plot(c_dist_crimes, type = "b")
```




```
influence = c_dist_crimes > 1
influence[influence == TRUE]
```

```
##    5    8   35   44
## TRUE TRUE TRUE TRUE
```

Points 5, 8, 35, and 44 in the figure above have Cook's distances greater than one, so they were classified as influence points.

b.) STEP UP: Step1:

```
library(broom)
mod1 = lm(expend ~ bad, data = expense)
summary(mod1)$r.squared
```

```
## [1] 0.6963839
```

```r
mod2 = lm(expend ~ crime, data = expense)
summary(mod2)$r.squared
```

```
## [1] 0.1118564
```

```r
mod3 = lm(expend ~ lawyers, data = expense)
summary(mod3)$r.squared
```

```
## [1] 0.9372789
```

```r
mod4 = lm(expend ~ employ, data = expense)
summary(mod4)$r.squared  # has greatest r^2 value
```

```
## [1] 0.9539745
```

```r
mod5 = lm(expend ~ pop, data = expense)
summary(mod5)$r.squared
```

```
## [1] 0.9073261
```

```r
summary(mod4)  #p-value significant
```

```
##
## Call:
## lm(formula = expend ~ employ, data = expense)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -636.04  -84.35   47.60  107.99 1124.70
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.167e+02  4.706e+01   -2.48   0.0166 *
## employ       4.681e-02  1.469e-03   31.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 257.4 on 49 degrees of freedom
## Multiple R-squared:  0.954,  Adjusted R-squared:  0.953
## F-statistic:  1016 on 1 and 49 DF,  p-value: < 2.2e-16
```

The model with *employ* variable yields the highest $R^2 = 0.954$ value. As it is also significant (p-value <2e-16), we add *employ* to our model.

Step2:

```r
mod6 = lm(expend ~ employ + bad, data = expense)
summary(mod6)$r.squared
```

```
## [1] 0.955097
```

```r
mod7 = lm(expend ~ employ + crime, data = expense)
summary(mod7)$r.squared
```

```
## [1] 0.9550501
```

```r
mod8 = lm(expend ~ employ + lawyers, data = expense)
summary(mod8)$r.squared
```

```
## [1] 0.9631745
```

```r
summary(mod8)
```

```
##
## Call:
## lm(formula = expend ~ employ + lawyers, data = expense)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -599.47  -94.43   36.01   91.98  936.55
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.107e+02  4.257e+01  -2.600  0.01236 *
## employ       2.971e-02  5.114e-03   5.810 4.89e-07 ***
## lawyers      2.686e-02  7.757e-03   3.463  0.00113 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 232.6 on 48 degrees of freedom
## Multiple R-squared:  0.9632, Adjusted R-squared:  0.9616
## F-statistic: 627.7 on 2 and 48 DF,  p-value: < 2.2e-16
```

```r
mod9 = lm(expend ~ employ + pop, data = expense)
summary(mod9)$r.squared
```

```
## [1] 0.95431
```

Adding variable *lawyers* results in maximum increase in R^2 and it is also significant from the results above.

Step3:

```r
mod10 = lm(expend ~ employ + lawyers + bad, data = expense)
summary(mod10)$r.squared  # r^2
```

```
## [1] 0.9638741
```

```r
summary(mod10)$coefficients[4, 4]  #p-value
```

```
## [1] 0.344957
```

```
mod11 = lm(expend ~ employ + lawyers + crime, data = expense)
summary(mod11)$r.squared
```

```
## [1] 0.9631881
```

```
summary(mod11)$coefficients[4, 4]
```

```
## [1] 0.8957792
```

```
mod12 = lm(expend ~ employ + lawyers + pop, data = expense)
summary(mod12)$r.squared
```

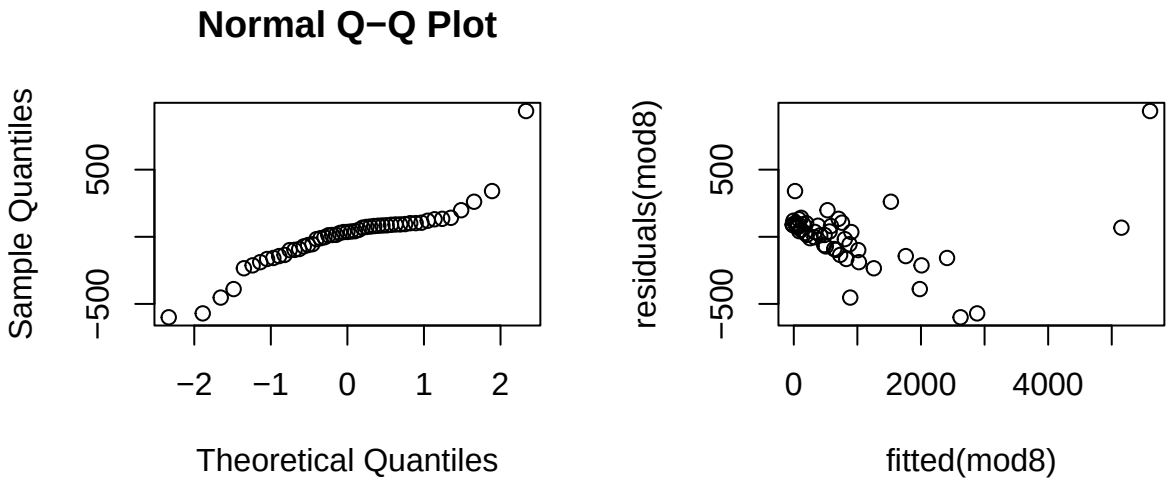
```
## [1] 0.9637326
```

```
summary(mod12)$coefficients[4, 4]
```

```
## [1] 0.399392
```

Adding any of the more variables does not cause a considerable increase in R^2 and all of them are insignificant (p-values>0.05). Hence, our best model is $mod8: expend = (-1.107e{+}02){+}(2.971e{-}02){*}employ{+}(2.686e{-}02) * lawyers + error$.

Diagnostics



Data does not look to be normal and the fitted-residuals plot seems to show some negative collinearity.

```
# Checking for collinearity
library(car)
```

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

```
vif(mod8)
```

```
##   employ  lawyers
## 14.83915 14.83915
```

The Vif values very high. Hence, there might be slight collinearity between the variables (which we had also observed from our graphical analysis).

c) Prediction interval for a (hypothetical) state with bad=50, crime=5000, lawyers=5000, employ=5000 and pop=5000.

```
newxdata = data.frame(bad = 50, crime = 5000, employ = 5000,
    lawyers = 5000, pop = 5000)
predict(mod8, newxdata, interval = "prediction", level = 0.95)
```

```
##        fit       lwr      upr
## 1 172.2098 -302.9307 647.3504
```

We can improve this interval by taking the confidence-interval as it is narrower than the prediction interval since the error is not taken into account in CI.

```
newxdata = data.frame(bad = 50, crime = 5000, employ = 5000,
    lawyers = 5000, pop = 5000)
predict(mod8, newxdata, interval = "confidence", level = 0.95)
```

```
##        fit      lwr      upr
## 1 172.2098 88.32196 256.0977
```

d) LASSO

```
expense = read.table("expensescrime.txt", header = TRUE)
library(glmnet)
```

```
## Loading required package: Matrix
```

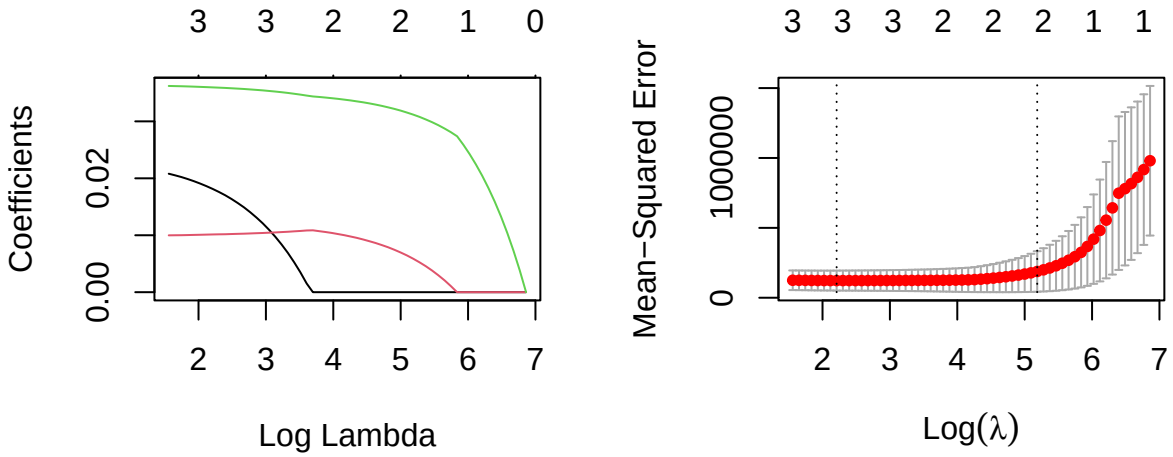
```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-6
```

```
expense = expense[, -1]
x = as.matrix(expense[, -1])  # remove the response variable
y = as.double(as.matrix(expense[, 1]))  # only the response variable expend
set.seed(1)
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]  # data to train
x.test = x[-train, ]
y.test = y[-train]  # data to test
lasso.mod = glmnet(x.train, y.train, alpha = 1)  # lasso model
lasso.cv = cv.glmnet(x.train, y.train, alpha = 1, type.measure = "mse")
par(mfrow = c(1, 2))
plot(lasso.mod, label = T, xvar = "lambda")  #have a look at the lasso path
plot(lasso.cv)  # the best lambda by cross-validation
```




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

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) 1.098690e+02
## bad         .
## crime       .
## lawyers     5.876409e-03
## employ      3.118708e-02
## pop         .
```

The LASSO regression method chooses *lawyers* and *employ* as the relevant variables. The coefficients can be observed in the output above.

```
# Prediction by using the linear model first fit linear
# model on the train data

y.predict.lm = predict(mod8, newdata = expense[-train, ])  # predict for the test rows
mse.lm = mean((y.test - y.predict.lm)^2)
mse.lm  # prediction quality by the linear model
```

```
## [1] 78414.05
```

```
lasso.pred = predict(lasso.mod, s = lambda.1se, newx = as.matrix(x.test))
mse2.lasso = mean((y.test - lasso.pred)^2)
mse2.lasso
```

```
## [1] 311977
```

When we compare the mean-squared-errors of the Lasso model above with our linear model from b.($mod8$), lasso yields a better mse.

Exercise 3

```
titanic = read.table(file = "titanic.txt", header = TRUE)
titanic_new <- na.omit(titanic)
```

The Age variable in the data has some 557 missing values. NA values have been omitted from the data.

a.)Graphics and tables

The tables below shows the total numbers of individuals for each combination of levels of Passenger Class and Sex, combination of Survival and Sex and number of survivals in terms of Passenger Class and Sex respectively.

```
tot = xtabs(~PClass + Sex, data = titanic)
knitr::kable(tot)
```

	female	male
1st	143	179
2nd	107	173

	female	male
3rd	212	499

```
tot.se = xtabs(~Survived + Sex, data = titanic)
knitr::kable(tot.se)
```

	female	male
0	154	709
1	308	142

```
tot.s = xtabs(Survived ~ PClass + Sex, data = titanic)
knitr::kable(tot.s)
```

	female	male
1st	134	59
2nd	94	25
3rd	80	58

Logistic Regression Model

```
titanic_new$PClass = as.factor(titanic_new$PClass)
titanic_new$Sex = as.factor(titanic_new$Sex)

model_titanic = glm(Survived ~ PClass + Age + Sex, data = titanic_new,
    family = binomial)
summary(model_titanic)$coefficients
```

```
##                Estimate  Std. Error    z value     Pr(>|z|)
## (Intercept)  3.75966210 0.397567324   9.456668 3.179129e-21
## PClass2nd   -1.29196240 0.260075781  -4.967638 6.777324e-07
## PClass3rd   -2.52141915 0.276656805  -9.113888 7.948131e-20
## Age         -0.03917681 0.007616218  -5.143868 2.691392e-07
## Sexmale     -2.63135683 0.201505379 -13.058494 5.684093e-39
```

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

```
## Single term deletions
##
## Model:
## Survived ~ PClass + Age + Sex
##        Df Deviance    AIC     LRT  Pr(>Chi)
## <none>      695.14 705.14
## PClass  2   795.59 801.59 100.445 < 2.2e-16 ***
## Age     1   723.59 731.59  28.454 9.595e-08 ***
## Sex     1   909.92 917.92 214.776 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we can see that all predictors are significant. The odds is the probablity of success divided by the probablity of failure. From the summary table we can see that the odds of survival for an individual is $exp(3.7597 + PClass2nd * -1.2920 + PClass3rd * -2.5214 + Sexmale * -2.6314 + Age * -0.0392)$.

For example, odds of survival for a 26 year old middle class male and female passengers

```
male_middle_class = exp(3.7597 + 1 * -1.292 + 0 * -2.5214 + 1 *
    -2.6314 + 26 * -0.0392)
female_middle_class = exp(3.7597 + 1 * -1.292 + 0 * -2.5214 +
    0 * -2.6314 + 26 * -0.0392)

male_middle_class
```

```
## [1] 0.3063889
```

```
female_middle_class
```

```
## [1] 4.256725
```

From the example above, 2nd PClass female has higher odds of survival compared to 2nd PClass male.

b)

```
model_1 <- glm(Survived ~ Age * Sex, data = titanic_new, family = binomial)
summary(model_1)$coefficients
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept)  0.30110864 0.29897816  1.007126 3.138743e-01
## Age          0.02935105 0.01007529  2.913174 3.577757e-03
## Sexmale     -0.59985816 0.40804956 -1.470062 1.415450e-01
## Age:Sexmale -0.06571791 0.01368620 -4.801765 1.572730e-06
```

```
model_2 <- glm(Survived ~ Age * PClass, data = titanic_new, family = binomial)
summary(model_2)$coefficients
```

```
##                   Estimate  Std. Error    z value     Pr(>|z|)
## (Intercept)    1.922979534 0.436245818  4.4080183 1.043207e-05
## Age           -0.035837742 0.009955455 -3.5998095 3.184504e-04
## PClass2nd     -0.744276585 0.571547192 -1.3022137 1.928433e-01
## PClass3rd     -2.290071973 0.540572606 -4.2363818 2.271504e-05
## Age:PClass2nd -0.013209098 0.015868842 -0.8323921 4.051877e-01
## Age:PClass3rd  0.004642132 0.015941232  0.2912028 7.708962e-01
```

To investigate the interaction of predictor Age with PClass, and the interaction of Age with Sex, two logistic regression models were created above. From these models we can see that there is significant interaction between Age and Sex, and there is no significant interaction between Age and PClass. Now we create a model with the Age and Sex interaction

```
model_3 <- glm(Survived ~ PClass + Age * Sex, data = titanic_new,
    family = binomial)
summary(model_3)$coefficients
```

```
##                Estimate Std. Error    z value     Pr(>|z|)
## (Intercept)  2.75656302 0.43764171  6.2986753 3.002001e-10
## PClass2nd   -1.54336652 0.28735776 -5.3708886 7.834962e-08
## PClass3rd   -2.65398052 0.29142296 -9.1069712 8.471384e-20
## Age          0.00244348 0.01140798  0.2141903 8.303986e-01
## Sexmale     -0.50818658 0.44251492 -1.1484055 2.508012e-01
## Age:Sexmale -0.07559126 0.01500877 -5.0364712 4.741923e-07
```

Now Age and sex are not significant anymore, therefore the final model would only consist of their interaction as an estimator. Then we end-up with the variables PClass and interaction of Age and Sex. Therfore, the chosen final model is shown below: $final\_model$.

```
final_model <- glm(Survived ~ PClass + Age:Sex, data = titanic_new,
    family = binomial)
summary(final_model)$coefficients
```

```
##                   Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept)    2.50401090 0.377774141  6.628328 3.395105e-11
## PClass2nd     -1.58098832 0.288018409 -5.489192 4.037768e-08
## PClass3rd     -2.66612966 0.292351736 -9.119596 7.540503e-20
## Age:Sexfemale  0.01080822 0.008942977  1.208570 2.268280e-01
## Age:Sexmale   -0.08022407 0.009173972 -8.744747 2.235162e-18
```

Then the estimate for the probability of survival for each combination of levels of the factors PClass and Sex for a person of age 55:

```
age <- 55
classes <- unique(as.character(titanic$PClass))
sexes <- unique(as.character(titanic$Sex))
new_data <- data.frame(PClass = rep(classes, length(sexes)),
    Sex = rep(sexes, each = length(classes)), Age = age)
# predict survival using final_model
results <- predict(final_model, new_data, type = "response")
# combine new_data and results into final data frame
final <- cbind(new_data, Survival = round(results, 3))
# print table
knitr::kable(final)
```

PClass	Sex	Age	Survival
1st	female	55	0.957
2nd	female	55	0.820
3rd	female	55	0.606
1st	male	55	0.129
2nd	male	55	0.030
3rd	male	55	0.010

c) To predict the survival status we create a logistic regression model on our given data-set then use this model to make predictions for unseen data-points. For a new data vector X, we can predict its success probability $\hat{P} = \frac{1}{1+e^{-x^T\hat{\theta}}}$. Now, we use $\hat{P}$ to predict the new survival status by comparing with some threshold $p_0 \in [0, 1]$. We can use accuracy accuracy as a quality measure for our prediction. **d)** The Fischer test is appropriate for investigating the effect of sex on survival since the data is in a 2x2 contingency table.

```
# contingency table and Fisher's exact test for Sex and
# Survived
tab1 <- table(titanic$Survived, titanic$Sex)
tab1_margins <- addmargins(tab1)
knitr::kable(tab1_margins)
```

	female	male	Sum
0	154	709	863
1	308	142	450
Sum	462	851	1313

```
fisher_res <- fisher.test(tab1)
fisher_res
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tab1
## p-value < 2.2e-16
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.07620521 0.13155709
## sample estimates:
## odds ratio
##  0.1003494
```

A Chi-square test is suitable for examining the influence of class on survival since the data is in a 2x3 contingency table and over 80% of the values have a value of more than 5.

```
# contingency table and chi-square test for PClass and
# Survived
tab2 <- table(titanic$Survived, titanic$PClass)
tab2_margins <- addmargins(tab2)
knitr::kable(tab2_margins)
```

	1st	2nd	3rd	Sum
0	129	161	573	863
1	193	119	138	450
Sum	322	280	711	1313

```
chi_res <- chisq.test(tab2)
chi_res
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab2
## X-squared = 172.3, df = 2, p-value < 2.2e-16
```

From the results of the chi-square and fisher tests it can be seen that both Sex and PClass have a significant effect on survival. **e)** The second approach, which involves using a contingency table with the Fisher and chi-squared test,is not necessarily wrong. However, the most appropriate method for answering this question is the contingency table with the Fisher and chi-squared test. This approach is suitable because it can help to determine whether the Survived variable is independent of the gender and class variables.

The choice between using the contingency table with the chi-squared test or logistic regression depends on the problem. Logistic regression is advantageous when we want to compute a probability, in this case, the probability that a person with a certain age, class, and gender will survive. On the other hand, the chi-squared test can be used to determine whether the reason a person survived is independent of their gender or class.

Exercise 4

```
coup_data = read.table(file = "coups.txt", header = TRUE)
```

a) Poisson regression

```
coup_data$pollib = as.factor(coup_data$pollib)
coupglm = glm(miltcoup ~ oligarchy + pollib + parties + pctvote +
    popn + size + numelec + numregim, family = poisson, data = coup_data)
summary(coupglm)
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties + pctvote +
##     popn + size + numelec + numregim, family = poisson, data = coup_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5075  -0.9533  -0.3100   0.4859   1.6459
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.2334274  0.9976112  -0.234  0.81500
## oligarchy    0.0725658  0.0353457   2.053  0.04007 *
## pollib1     -1.1032439  0.6558114  -1.682  0.09252 .
## pollib2     -1.6903057  0.6766503  -2.498  0.01249 *
## parties      0.0312212  0.0111663   2.796  0.00517 **
## pctvote      0.0154413  0.0101027   1.528  0.12641
## popn         0.0109586  0.0071490   1.533  0.12531
## size        -0.0002651  0.0002690  -0.985  0.32444
## numelec     -0.0296185  0.0696248  -0.425  0.67054
## numregim     0.2109432  0.2339330   0.902  0.36720
## ---
## 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: 113.06
##
## Number of Fisher Scoring iterations: 5
```

Only *oligarchy*, *pollib*, and *parties* variables show a significant effect with a $p - value < 0.05$, for the pollib factor variable we also see that only *pollib*2 level (full civil rights) has a significant influence. Additionally, *numelec* and the number of regime (*numregim*) seem to have the least influence on the number of coups.
b) Step down Step 1: Explanatory variable numelec has the highest p-value in the above model. So our first step in the step-down process is removing numelec.

```
coup_1 = glm(miltcoup ~ oligarchy + pollib + parties + pctvote +
    popn + size + numregim, family = poisson, data = coup_data)
summary(coup_1)$coefficients
```

```
##                 Estimate   Std. Error    z value    Pr(>|z|)
## (Intercept) -0.4577458289 0.8602344719 -0.5321175 0.594644608
## oligarchy    0.0812015376 0.0288154207  2.8179890 0.004832547
## pollib1     -0.9642975542 0.5620939337 -1.7155452 0.086245314
## pollib2     -1.5149509438 0.5269441006 -2.8749747 0.004040599
## parties      0.0293409468 0.0103100564  2.8458571 0.004429207
## pctvote      0.0139115305 0.0094653971  1.4697250 0.141636255
## popn         0.0099592030 0.0067248724  1.4809505 0.138619769
## size        -0.0002687704 0.0002686512 -1.0004436 0.317095873
## numregim     0.1804415213 0.2241166271  0.8051233 0.420748524
```

Step2: The next highest insignificant p-value is *numeregim*. So removing that-

```
coup_2 = glm(miltcoup ~ oligarchy + pollib + parties + pctvote +
    popn + size, family = poisson, data = coup_data)
summary(coup_2)$coefficients
```

```
##                 Estimate   Std. Error     z value    Pr(>|z|)
## (Intercept)  0.0419756813 0.5774100214  0.07269649 0.942047643
## oligarchy    0.0894950616 0.0270440021  3.30923882 0.000935500
## pollib1     -0.9673252804 0.5605601269 -1.72564054 0.084412101
## pollib2     -1.5321125526 0.5232779163 -2.92791365 0.003412448
## parties      0.0288170423 0.0102172799  2.82042211 0.004796051
## pctvote      0.0149215757 0.0093762023  1.59143065 0.111512690
## popn         0.0071646561 0.0056842444  1.26044124 0.207510232
## size        -0.0002579079 0.0002662008 -0.96884720 0.332621435
```

Step3: Remove *size* next.

```
coup_3 = glm(miltcoup ~ oligarchy + pollib + parties + pctvote +
    popn, family = poisson, data = coup_data)
summary(coup_3)$coefficients
```

```
##                Estimate  Std. Error    z value    Pr(>|z|)
## (Intercept) -0.231434917 0.528887463 -0.4375882 0.661684821
## oligarchy    0.083467586 0.025829007  3.2315445 0.001231231
## pollib1     -0.683589192 0.495822322 -1.3786979 0.167987918
## pollib2     -1.320568052 0.490268490 -2.6935609 0.007069322
## parties      0.029769711 0.010309890  2.8874907 0.003883281
## pctvote      0.013924684 0.009370609  1.4859956 0.137280288
## popn         0.005659313 0.005483446  1.0320723 0.302038232
```

Step4: Remove *popn*

```
coup_4 = glm(miltcoup ~ oligarchy + pollib + parties + pctvote,
    family = poisson, data = coup_data)
summary(coup_4)$coefficients
```

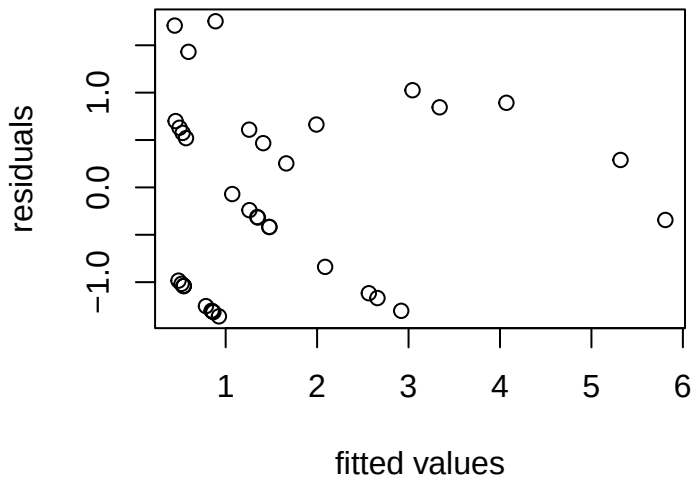
```
##                Estimate  Std. Error    z value     Pr(>|z|)
## (Intercept) -0.11649888 0.513750518 -0.2267616 8.206091e-01
## oligarchy    0.09471197 0.023183825  4.0852606 4.402738e-05
## pollib1     -0.62075614 0.487525525 -1.2732793 2.029190e-01
## pollib2     -1.31037384 0.489017399 -2.6796058 7.370891e-03
## parties      0.02574472 0.009552367  2.6951146 7.036443e-03
## pctvote      0.01205704 0.009071988  1.3290409 1.838345e-01
```

Step5:remove pctvote

```
coup_5 = glm(miltcoup ~ oligarchy + pollib + parties, family = poisson,
    data = coup_data)
summary(coup_5)
```

```
##
## Call:
## glm(formula = miltcoup ~ oligarchy + pollib + parties, family = poisson,
##     data = coup_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3609  -1.0407  -0.3153   0.6145   1.7536
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.207981   0.445679   0.467   0.6407
## oligarchy    0.091466   0.022563   4.054 5.04e-05 ***
## pollib1     -0.495414   0.475645  -1.042   0.2976
## pollib2     -1.112086   0.459492  -2.420   0.0155 *
## parties      0.022358   0.009098   2.458   0.0140 *
## ---
## 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: 107.63
##
## Number of Fisher Scoring iterations: 5
```

The final Poisson model contains the same three variables that were previously identified as significant in question 4a(oligarchy, pollib, and parties). Furthermore, only one of the *pollib* factor levels (pollib2) remains significant. All the remaining variables are significant. So $coup_5$ is our final model. **Resulting model**: $miltcoup = 0.207981 + 0.091466 * oligarchy - 0.495414 * pollib1 - 1.112086 * pollib2 + 0.022358 * parties$.



Diagnostics:

c) We calculate the overall average values for parties and oligarchy and predict the number of coups for all the three levels of political liberalization for those values.

```
avg_oligarchy = mean(coup_data$oligarchy)
avg_parties = mean(coup_data$parties)
new_data0 = data.frame(oligarchy = avg_oligarchy, pollib = "0",
    parties = avg_parties)
new_data1 = data.frame(oligarchy = avg_oligarchy, pollib = "1",
    parties = avg_parties)
new_data2 = data.frame(oligarchy = avg_oligarchy, pollib = "2",
    parties = avg_parties)
```

```
predict(coup_5, new_data0, type = "response")[[1]]
```

```
## [1] 2.908352
```

```
predict(coup_5, new_data1, type = "response")[[1]]
```

```
## [1] 1.772113
```

```
predict(coup_5, new_data2, type = "response")[[1]]
```

```
## [1] 0.9564757
```

So, $pollib = 0$, i.e. no civil rights for political expression has the highest probability of a military coup, with limited civil rights($pollib = 1$) having the next highest chance and the full civil rights($pollib = 1$) with least probability of military coups.