Problem 8: UK Election Results, 2015-2019

Rakesh Vishwabrahmana

```
# Require Packages
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(GGally)

## Warning: package 'GGally' was built under R version 4.1.2

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(ggfortify)
library(plyr)

## -----------------------------------------------------------------------------
----

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first,
then dplyr:
## library(plyr); library(dplyr)

## -----------------------------------------------------------------------------
----

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact

library(olsrr)

## Warning: package 'olsrr' was built under R version 4.1.2

##
## Attaching package: 'olsrr'

## The following object is masked from 'package:datasets':
##
##     rivers

library(car)

## Warning: package 'car' was built under R version 4.1.2

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

Question: 1

Solution:

(a)

In the following r chunk, we load all relevant CSV files into R data frame variables and convert string variable as factor.

```
GE2015 <- read_csv("GE2015-results.csv")

## Rows: 650 Columns: 28

## -- Column specification --------------------------------------------------------
------
## Delimiter: ","
## chr (11): ons_id, ons_region_id, constituency_name, county_name,
region_name...
## dbl (17): electorate, valid_votes, invalid_votes, majority, con, lab, ld,
uk...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

GE2015 <- GE2015%>%
  mutate_if(is.character,as.factor)


GE2017 <- read_csv("GE2017-results.csv")

## Rows: 650 Columns: 29

## -- Column specification --------------------------------------------------------
------
## Delimiter: ","
## chr  (10): ons_id, ons_region_id, constituency_name, county_name,
region_nam...
## dbl  (18): electorate, valid_votes, invalid_votes, majority, con, lab, ld,
u...
## dttm  (1): declaration_time

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

GE2017 <- GE2017%>%
  mutate_if(is.character,as.factor)


GE2019 <- read_csv("GE2019-results.csv")

## Rows: 650 Columns: 32

## -- Column specification --------------------------------------------------------
------
## Delimiter: ","
## chr  (13): ons_id, ons_region_id, constituency_name, county_name,
region_nam...
## dbl  (18): electorate, valid_votes, invalid_votes, majority, con, lab, ld,
b...
## dttm  (1): declaration_time

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

GE2019 <- GE2019 %>%
  mutate_if(is.character,as.factor)


demographics <- read_csv("demographics.csv")
```

```
## Rows: 650 Columns: 13

## -- Column specification -------------------------------------------------------
------
## Delimiter: ","
## chr  (2): ons_id, constituency
## dbl (11): income, age.0.15, age.65.over, foreignborn, employment,
outofwork,...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

demographics <- demographics %>%
  mutate_if(is.character,as.factor)
```

(b)

We add an extra categorical column **year** using the year of the election 2015, 2017 and 2019 to each results data frame.

```
GE2015 <- GE2015 %>%
  mutate(year= "2015")
GE2017 <- GE2017 %>%
  mutate(year= "2017")
GE2019 <- GE2019 %>%
  mutate(year= "2019")
```

(c)

We will merge all the three separate year-specific results data frames into a single data frame containing all results in the following chunk.

```
mrg_data <- plyr::rbind.fill(GE2015, GE2017, GE2019)
dim(mrg_data)

## [1] 1950   34
```

(d)

In the given chunk below we merge the combined results data frame together with the demographics data to produce a single data frame.

```
full_data<-left_join(mrg_data ,demographics,by = "ons_id" ,copy=TRUE)
```

(e)

We calculate the conservative vote share, then apply a logit transform to this value and create a new column named **con_share** of the data frame.

```
full_data <- full_data %>%
  mutate(con_share  = car::logit(con/valid_votes))
```

```
## Warning in car::logit(con/valid_votes): proportions remapped to (0.025,
0.975)
```

(f)

We select only relevant columns for the modeling purpose which will contain response and explanatory variable and define the object as final data frame.

```
new_data <-full_data %>%
  select(c(year,country_name, income, age.0.15, age.65.over,
           foreignborn,employment,outofwork, white,commute.car,
           commute.bike, health.good,health.bad, con_share))
```

(g)

We will filter out rows associated with *Northern Ireland* constituencies from the data set.

```
mod_data <- new_data %>%
  filter(country_name != "Northern Ireland") %>%
  mutate_if(is.character,as.factor) %>%
  drop_na()

dim(mod_data)

## [1] 1896   14
```

(h)

In the following chunk, we will use base function `summary()` to summarize variables in our final data frame.

```
summary(mod_data)

##    year                country_name      income          age.0.15
##  2015:632   England          :1599   Min.   :17900   Min.   :11.30
##  2017:632   Northern Ireland:   0   1st Qu.:21200   1st Qu.:17.40
##  2019:632   Scotland         : 177   Median :22600   Median :18.40
##             Wales            : 120   Mean   :23423   Mean   :18.63
##                                      3rd Qu.:24900   3rd Qu.:19.80
##                                      Max.   :44300   Max.   :30.40
##   age.65.over     foreignborn      employment       outofwork
##  Min.   : 5.5   Min.   : 2.00   Min.   :42.00   Min.   : 0.480
##  1st Qu.:14.6   1st Qu.: 4.60   1st Qu.:58.88   1st Qu.: 1.417
##  Median :16.8   Median : 7.50   Median :62.20   Median : 2.320
##  Mean   :16.8   Mean   :11.85   Mean   :61.78   Mean   : 2.627
##  3rd Qu.:19.4   3rd Qu.:13.57   3rd Qu.:65.80   3rd Qu.: 3.475
##  Max.   :32.2   Max.   :59.30   Max.   :74.60   Max.   :10.250
##      white         commute.car      commute.bike      health.good
##  Min.   :23.10   Min.   :10.61   Min.   : 0.200   Min.   :70.60
##  1st Qu.:85.95   1st Qu.:62.41   1st Qu.: 1.238   1st Qu.:78.70
##  Median :94.40   Median :69.44   Median : 1.960   Median :81.35
##  Mean   :88.04   Mean   :65.03   Mean   : 2.602   Mean   :81.11
```

```
##   3rd Qu.:97.60    3rd Qu.:74.46    3rd Qu.: 3.125    3rd Qu.:83.50  
##   Max.   :99.20    Max.   :83.33    Max.   :30.350    Max.   :89.00  
##     health.bad        con_share      
##   Min.   : 2.400   Min.   :-3.6636  
##   1st Qu.: 4.475   1st Qu.:-0.9117  
##   Median : 5.300   Median :-0.2304  
##   Mean   : 5.667   Mean   :-0.4054  
##   3rd Qu.: 6.600   3rd Qu.: 0.1800  
##   Max.   :11.600   Max.   : 1.1194
```

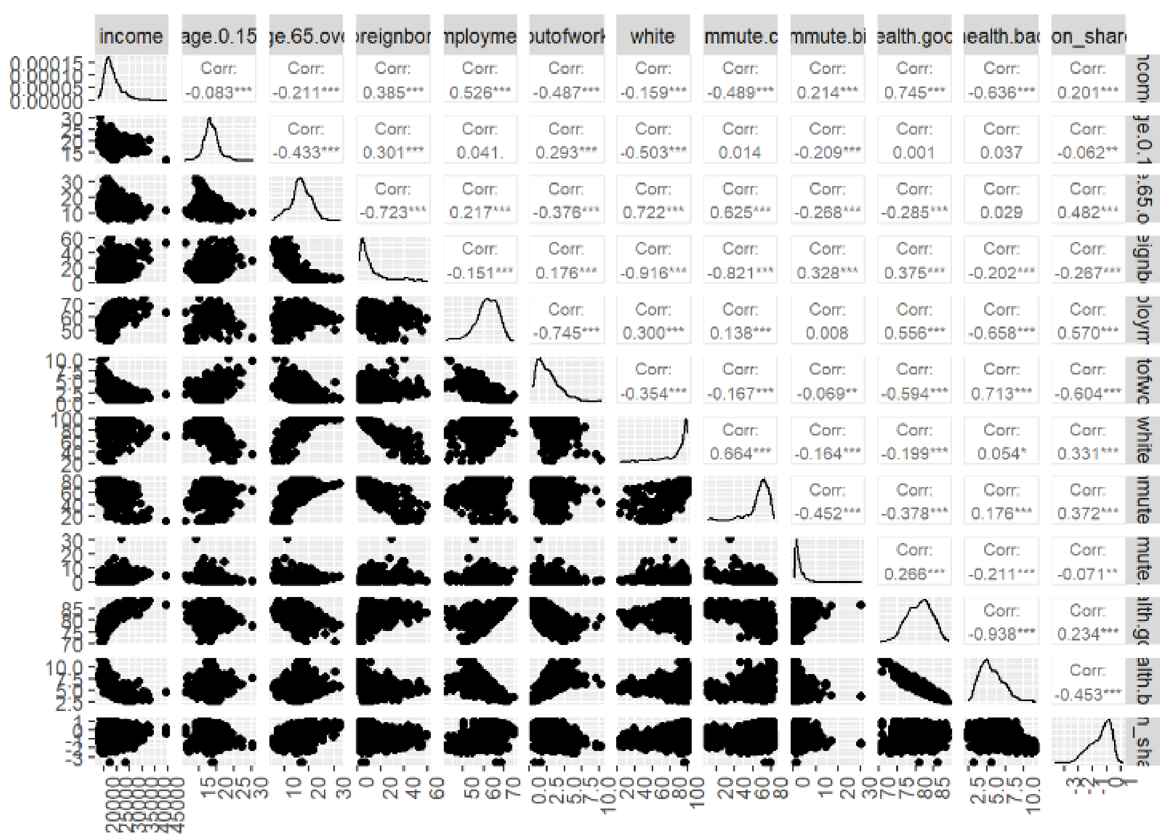
Question: 2

Solution:

The following R chunk below we will create a data frame with all the numeric variables and visualize there relationship using a pair matrix plot.

```
num_data <- mod_data %>%
  select_if(is.numeric)

ggpairs(num_data,upper = list(continuous = wrap("cor", size = 2.5))) +
theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Question: 3

Solution:

In the given chunk below we will give a table which will show whether the relationship with the response variable is positively correlated (+), not significantly correlated (o), or negatively correlated (-).

```
pos <- c("income", "age.65.over", "employment", "white", "commute.car",
"health.good")
nsig <- rep(NA, 6)
neg <- c("age.0.15","foreignborn", "outofwork", "commute.bike", "health.bad",
NA)
cor_tab <- data.frame(pos, nsig, neg)
names(cor_tab)<- c("+","o","-")
cor_tab

##             +  o            -
## 1      income NA     age.0.15
## 2 age.65.over NA  foreignborn
## 3  employment NA    outofwork
## 4       white NA commute.bike
## 5 commute.car NA   health.bad
## 6 health.good NA         <NA>
```

Question: 4

Solution:

In the following chunk below we will find the association between the response and categorical variables and find the group that predicts the highest and lowest value of the response variable.

```
cat_data <- mod_data %>%
  select(con_share,year,country_name)
mod1 <- aov(con_share~ year+country_name, data = cat_data)
summary(mod1)

##                Df Sum Sq Mean Sq F value Pr(>F)    
## year            2   39.2   19.59   45.23 <2e-16 ***
## country_name    2  179.6   89.81  207.37 <2e-16 ***
## Residuals    1891  819.0    0.43                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

cat_data%>%
  group_by(year) %>%
  summarise(ave_con_share=mean(con_share))

##   ave_con_share
## 1    -0.4054456
```

```
cat_data%>%
  group_by(country_name) %>%
  summarise(ave_con_share=mean(con_share))

##   ave_con_share
## 1    -0.4054456

data.frame(Highest = c("2019","England"), lowest=c("2015","Scotland"))

##   Highest   lowest
## 1    2019     2015
## 2 England Scotland
```

Question: 5

Solution:

```
# (a)
null_mod <- lm(con_share~1,data=mod_data)
# (b)
full_mod <- lm(con_share~.,data=mod_data)
# (c)
int.year_mod <- lm(con_share~.+year*.,data=mod_data)
# (d)
int.all_mod <- lm(con_share~.*.,data=mod_data)
# (e)
step_c <- step(int.year_mod, trace=0)
# (f)
step_d <- step(int.all_mod, trace=0)
```

Question: 6

Solution:

(a) Degrees of error freedom

```
df_e <- c(null_mod$df.residual,
          full_mod$df.residual,
          int.year_mod$df.residual,
          int.all_mod$df.residual,
          step_c$df.residual,
          step_d$df.residual)
```

(b) Degrees of model freedom

```
N <- dim(mod_data)[1]
df_mod <- c(N - null_mod$df.residual,
            N - full_mod$df.residual,
            N - int.year_mod$df.residual,
            N - int.all_mod$df.residual,
            N - step_c$df.residual,
            N - step_d$df.residual)
```

(c) Multiple R-squared

```
r2 <- c(summary(null_mod)$r.squared,
        summary(full_mod)$r.squared,
        summary(int.year_mod)$r.squared,
        summary(int.all_mod)$r.squared,
        summary(step_c)$r.squared,
        summary(step_d)$r.squared)
```

(d) Adjusted R-squared

```
adj_r2 <- c(summary(null_mod)$adj.r.squared,
            summary(full_mod)$adj.r.squared,
            summary(int.year_mod)$adj.r.squared,
            summary(int.all_mod)$adj.r.squared,
            summary(step_c)$adj.r.squared,
            summary(step_d)$adj.r.squared)
```

(e) Residual standard error

```
se_e <- c(summary(null_mod)$sigma,
          summary(full_mod)$sigma,
          summary(int.year_mod)$sigma,
          summary(int.all_mod)$sigma,
          summary(step_c)$sigma,
          summary(step_d)$sigma)
```

(f) Akaike information criterion

```
aic <- c(AIC(null_mod),
         AIC(full_mod),
         AIC(int.year_mod),
         AIC(int.all_mod),
         AIC(step_c),
         AIC(step_d))
```

(h) PRESS

```
press <- c(olsrr::ols_press(null_mod),
           olsrr::ols_press(full_mod),
           olsrr::ols_press(int.year_mod),
           olsrr::ols_press(int.all_mod),
           olsrr::ols_press(step_c),
           olsrr::ols_press(step_d))

data.frame(df_e, df_mod, r2, adj_r2, se_e, aic, press)

##   df_e df_mod        r2    adj_r2      se_e      aic     press
## 1 1895      1 0.0000000 0.0000000 0.7400362 4242.010 1038.8991
## 2 1880     16 0.7262950 0.7241111 0.3887049 1815.354  289.2403
## 3 1854     42 0.7614822 0.7562076 0.3653954 1606.451  259.4941
## 4 1777    119 0.8283172 0.8169167 0.3166484 1137.056  200.2715
## 5 1866     30 0.7611447 0.7574326 0.3644762 1585.132  256.5791
## 6 1808     88 0.8273494 0.8190415 0.3148056 1085.713  195.3835
```

Question: 7

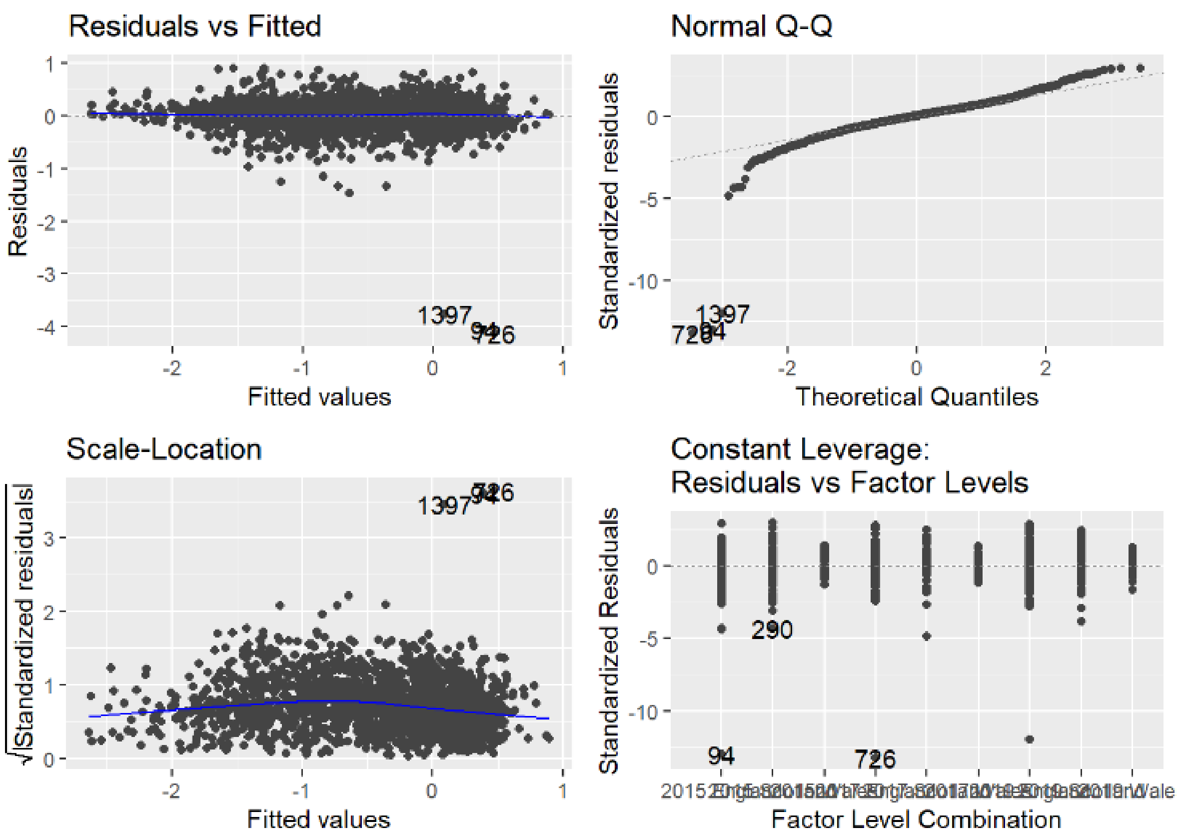
Solution:

From the above table we observed that the *step_d* might be the best model depending on the model metrics. However, the full model also perform well to explain the variation of Conservative vote share. The model with interaction can improve a bit to explain the variation of Conservative vote share.

Question: 8

Solution:

```
autoplot(step_d)
```



From the above diagnostic plot, we observed that the residual are approximately normally distributed but three observations are clearly outlier.

Question: 9

Solution:

```
summary(step_d)$coefficients[c("country_nameScotland","foreignborn","country_
nameScotland:health.bad"), ]

##                                    Estimate Std. Error   t value
Pr(>|t|)
```

```
## country_nameScotland            -56.168179  7.4680828 -7.521097 8.507891e-
14
## foreignborn                      -1.696258  0.2450373 -6.922450 6.143609e-
12
## country_nameScotland:health.bad   1.228138  0.1361882  9.017947 4.760267e-
19
```

The variables country_name, foreignborn , and the interaction country_name with health.bad are highly significant to reduce the variance in the “best” model. The proportion of vote in Scotland is very much less than the England. For *foreignborn*, we observed same scenarios like Scotland. Nevertheless, the interaction between Scotland and health.bad have positive association with proportion of vote, i.e, the residence in Scotland with poor health condition are likely to give their vote Conservative party.