Problem set 7

Your name here

Due 11/15/2022 at 5pm

NOTE1: Start with the file `ps7_2022.Rmd` (available from the github repository at https://github.com/UChicago-pol-methods/IntroQSS-F22/tree/main/assignments). Modify that file to include your answers. Make sure you can "knit" the file (e.g. in RStudio by clicking on the `Knit` button). Submit both the Rmd file and the knitted PDF via Canvas

To make the results of your knitted problem sets comparable, set the seed to (arbitrarily chosen) 60637:

```
# keep this code as-is
set.seed(60637)
```

Question 1: Best linear predictor vs OLS

Consider the following joint PMF:

$$f_{X,Y}(x,y) = \begin{cases} 1/3 & x=0, y=0 \\ 1/6 & x=0, y=1 \\ 1/6 & x=1, y=1 \\ 1/3 & x=1, y=2 \\ 0 & \text{otherwise} \end{cases}$$

(1a) What are the coefficients (slope and intercept) of the best linear predictor (BLP) of Y given X? (Show your work, which will require computing $\mathrm{E}[X]$, $\mathrm{E}[Y]$, $\mathrm{V}[X]$, and $\mathrm{Cov}[X,Y]$.)

Answer:

- $\mathrm{E}[X] = \frac{1}{2} \times 0 + \frac{1}{2} \times 1 = \frac{1}{2}$
- $\mathrm{E}[Y] = \frac{1}{3} \times 0 + \frac{1}{3} \times 1 + \frac{1}{3} \times 2 = 1$
- $\mathrm{V}[X] = \mathrm{E}[X^2] - \mathrm{E}[X]^2 = \frac{1}{2} - \frac{1}{4} = \frac{1}{4}$
- $\mathrm{Cov}[X,Y] = \mathrm{E}[XY] - \mathrm{E}[X]\mathrm{E}[Y] = \left(\frac{1}{6} \times 1 + \frac{1}{3} \times 2\right) - \frac{1}{2} = \frac{5}{6} - \frac{3}{6} = \frac{1}{3}$

Therefore

- slope (β): $\frac{\mathrm{Cov}[X,Y]}{\mathrm{V}[X]} = \frac{1/3}{1/4} = \frac{4}{3}$
- intercept (α): $\mathrm{E}[Y] - \beta \mathrm{E}[X] = 1 - \frac{4}{3}\frac{1}{2} = \frac{1}{3}$

(1b) What is the prediction of the BLP at $X = 1$? Confirm that this is the same as $E[Y|X=1]$.

Answer: $\alpha + \beta X$ where $X = 1$ is $\frac{5}{3}$.

$\mathrm{E}[Y|X=1] = \frac{1/6}{1/6+1/3} \times 1 + \frac{1/3}{1/6+1/3} \times 2 = \frac{5}{3}$

(1c) Make a tibble with the same joint distribution of `x` and `y` as the joint PMF above. Regress `y` on `x` in this dataset, present the results in a regression table using the `huxreg()` command in the `huxtable` package, and confirm check that you recover the coefficients of the BLP.

Answer:

```
dat <- tibble(x = c(0, 0, 0, 1, 1, 1), y = c(0, 0, 1, 1, 2, 2))
lm(y ~ x, data = dat) |>
  huxtable::huxreg()
```

	(1)
(Intercept)	0.333
	(0.333)
x	1.333 *
	(0.471)
N	6
R2	0.667
logLik	-4.001
AIC	14.003

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.

(1d) Look up the `slice_sample()` command (part of the `dplyr` package, which is part of `tidyverse`). Draw a sample of size 100 (with replacement) from the tibble you created in (1c) and again regress `y` on `x` and store the result. Do the same again but make the sample size 1000. Use `huxreg()` in the `huxtable` package to display the regression results side by side. Comment about the two sets of results and how they relate to the BLP.

Answer:

```
sampled_100 <- dat |>
  slice_sample(n = 100, replace = T)
lm_100 <- lm(y ~ x, data = sampled_100)
sampled_1000 <- dat |>
  slice_sample(n = 1000, replace = T)
lm_1000 <- lm(y ~ x, data = sampled_1000)

huxtable::huxreg(lm_100, lm_1000)
```

Question 2: OLS mechanics

Load the data on presidential elections from the course github:

```
pres <- read_csv("https://raw.githubusercontent.com/UChicago-pol-methods/IntroQSS-F22/main/data/pres_dat
```

```
## Rows: 19 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): year, deminc, incvote, q2gdp, juneapp
##
## 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.
```

	(1)	(2)
(Intercept)	0.386 ***	0.331 ***
	(0.063)	(0.021)
x	1.335 ***	1.324 ***
	(0.096)	(0.030)
N	100	1000
R2	0.663	0.662
logLik	-66.550	-670.239
AIC	139.099	1346.477

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.

2a) Regress the incumbent party's vote share (`incvote`) on the president's approval rating in June (`juneapp`). Store the result and report it using `huxtable::huxreg()`.

Answer:

```
pres_lm <- lm(incvote ~ juneapp, data = pres)
huxtable::huxreg(pres_lm)
```

	(1)
(Intercept)	51.037 ***
	(0.773)
juneapp	0.165 ***
	(0.030)
N	18
R2	0.655
logLik	-45.385
AIC	96.770

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.

2b) Write a function that computes the mean squared residual from a linear prediction of `incvote` based on `juneapp` given a slope and intercept. You may want to start by writing code that takes the `pres` dataset, generates predicted `incvote` given a slope and intercept, and computes the mean squared residual. Then wrap this in a function. The arguments to your function should be a `slope` and an `intercept`. Make sure the function returns a numeric value – you might need to use `as.numeric()` to convert the raw result of your code into a number. Use the function to compute the mean squared residual we obtain when we predict `incvote` using `juneapp` with the intercept you estimated in (2a) and a slope of .1.

Answer:

```r
msr_func <- function(slope, intercept){
  pres |>
    mutate(predicted = intercept + slope*juneapp,
           resid = incvote - predicted) |>
    summarize(mean(resid^2, na.rm = T)) |>
    as.numeric()
}
msr_func(slope = .1, intercept = coef(pres_lm)[1])
```

```
## [1] 11.88691
```

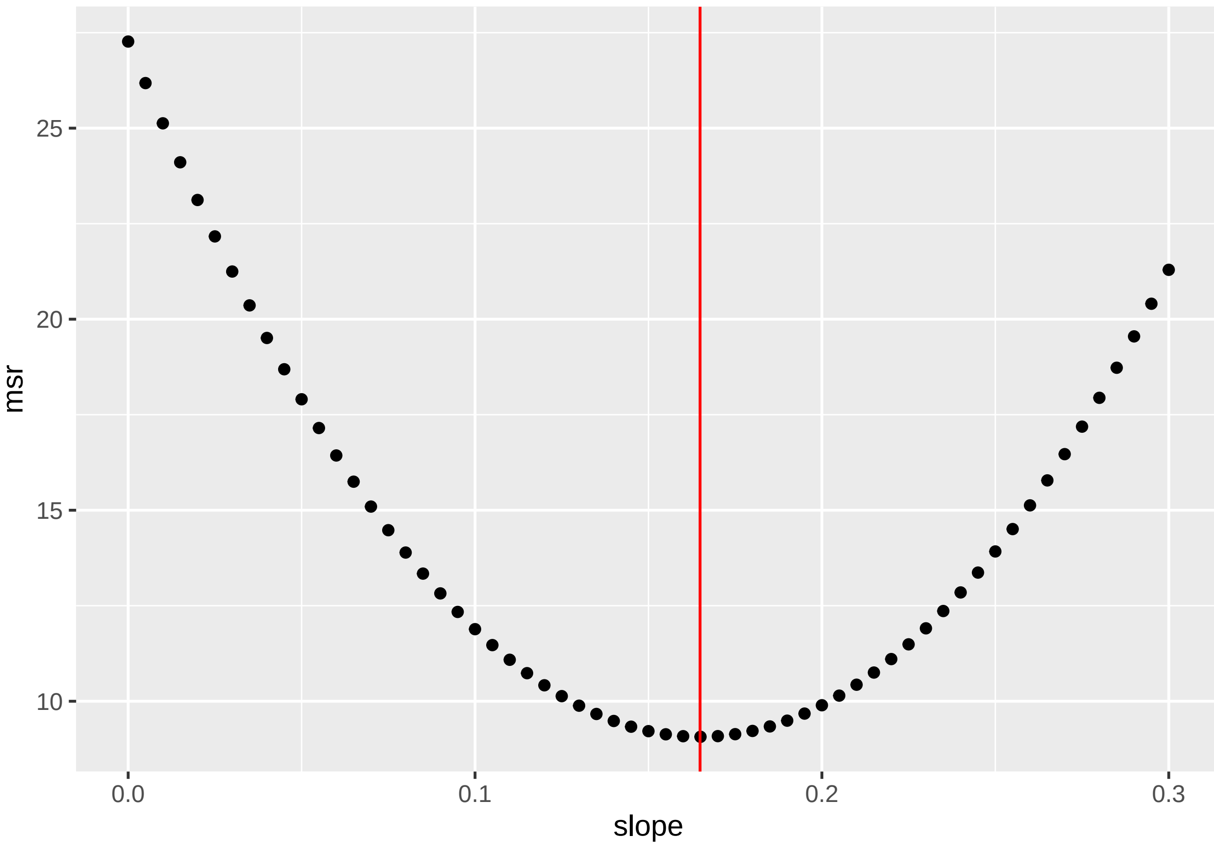
2c) Using the function you wrote, compute the mean squared residual for a sequence of slopes between 0 and .3 (by increments of .005) and again using the intercept you computed in (2a). (Hint: you could use `map_dbl`, `map2_dbl`, `sapply`, or a for-loop to do this.) Plot the mean squared residual for each value of the slope, and add a red vertical line at the OLS slope you computed in (2a).

Answer:

```r
# this is the sequence of slopes we will use
slopes <- seq(0, .3, by = .005)

# this is how to compute msrs using map_dbl
data_for_plot_map <- tibble(slope = slopes,
                            msr = map_dbl(slopes, msr_func, intercept = coef(pres_lm)[1]))

# now make the plot
data_for_plot_map |>
  ggplot(aes(x = slope, y = msr)) +
  geom_point() +
  geom_vline(xintercept = coef(pres_lm)[2], col = "red")
```



Below I show how to generate the same data using other approaches to iteration:

```
# 2. map2_dbl version
data_for_plot_map_2 <- tibble(slope = slopes,
                              intercept = coef(pres_lm)[1]) |>
  mutate(msr = map2_dbl(slopes, intercept, msr_func))

# 3. sapply version
data_for_plot_map_3 <- tibble(slope = slopes,
                              msr = sapply(slopes, msr_func, intercept = coef(pres_lm)[1]))

# 4. for-loop version
msrs <- rep(0, length(slopes))
for(i in 1:length(slopes)){
  msrs[i] <- msr_func(slope = slopes[i], intercept = coef(pres_lm)[1])
}
data_for_plot_map_4 <- tibble(slope = slopes,
                              msr = msrs)
```

Question 3: Interpretation of regression coefficients

The CSV at https://andy.egge.rs/data/brexit/brexit_data.csv contains results of the 2016 UK Brexit referendum by local authority (collected from the Electoral Commission website) and 2011 census data. It was gathered by Claire Peacock.

3a) Load the data.

```r
brexit <- read_csv("https://andy.egge.rs/data/brexit/brexit_data.csv")
```

```
## Rows: 382 Columns: 61
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): Area, Region_Code, Region, Area_Code
## dbl (57): Electorate, ExpectedBallots, VerifiedBallotPapers, Percent_Turnout...
##
## 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.
```

3b) Use `group_by()` and `summarize()` to make a table showing, for each `Region`, (i) the mean of `Percent_Leave` and (ii) the number of local authorities. (You may find the `n()` function useful.) Store the table for later use and display it below.

```r
by_region <- brexit |>
  group_by(Region) |>
  summarize(leave_pct = mean(Percent_Leave), n = n())
by_region
```

```
## # A tibble: 12 x 3
##    Region                   leave_pct     n
##    <chr>                        <dbl> <int>
##  1 East                          57.0    47
##  2 East Midlands                 59.6    40
##  3 London                        39.1    33
##  4 North East                    59.5    12
##  5 North West                    55.9    39
##  6 Northern Ireland              44.2     1
##  7 Scotland                      39.1    32
##  8 South East                    52.2    67
##  9 South West                    52.4    38
## 10 Wales                         53.3    22
## 11 West Midlands                 60.3    30
## 12 Yorkshire and The Humber      58.6    21
```

3c) (Law of iterated expectations applied to a sample) Compute the mean of `Percent_Leave` in this dataset in two ways: (i) unconditionally (the analogue of $E[Y]$) and (ii) as the weighted average of the region averages (the analogue of $E[E[Y \mid X]]$).

```r
brexit |>
  summarize(mean(Percent_Leave))
```

```
## # A tibble: 1 x 1
##   `mean(Percent_Leave)`
##                   <dbl>
## 1                  53.0
```

```r
by_region |>
  mutate(prop = n/sum(n)) |>
  summarize(sum(leave_pct*prop))
```

```
## # A tibble: 1 x 1
##   `sum(leave_pct * prop)`
##                     <dbl>
## 1                    53.0
```

3d) Regress `Percent_Leave` on `Region`. Output the result using `huxtable::huxreg()`.

```
lm(Percent_Leave ~ Region, data = brexit) |>
  huxtable::huxreg(error_pos = "right") # putting standard errors on the right
```

	(1)	
(Intercept)	56.963 ***	(1.207)
RegionEast Midlands	2.612	(1.780)
RegionLondon	-17.872 ***	(1.880)
RegionNorth East	2.515	(2.677)
RegionNorth West	-1.048	(1.793)
RegionNorthern Ireland	-12.738	(8.363)
RegionScotland	-17.826 ***	(1.897)
RegionSouth East	-4.792 **	(1.575)
RegionSouth West	-4.584 *	(1.805)
RegionWales	-3.615	(2.138)
RegionWest Midlands	3.352	(1.934)
RegionYorkshire and The Humber	1.686	(2.172)
N	382	
R2	0.419	
logLik	-1343.237	
AIC	2712.474	

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.

```
# so it fits better in the output
```

3e) Based on your regression, what is the predicted support for Leave in a local authority in London? Compare your answer to the average support for Leave in London authorities in the data.

Answer: The regression prediction is 56.963 - 17.872 = 39.091.

The average support in London local authorities in the data should be the same:

```
brexit |>
  filter(Region == "London") |>
  summarize(mean(Percent_Leave))
```

```
## # A tibble: 1 x 1
##   `mean(Percent_Leave)`
##                   <dbl>
## 1                  39.1
```

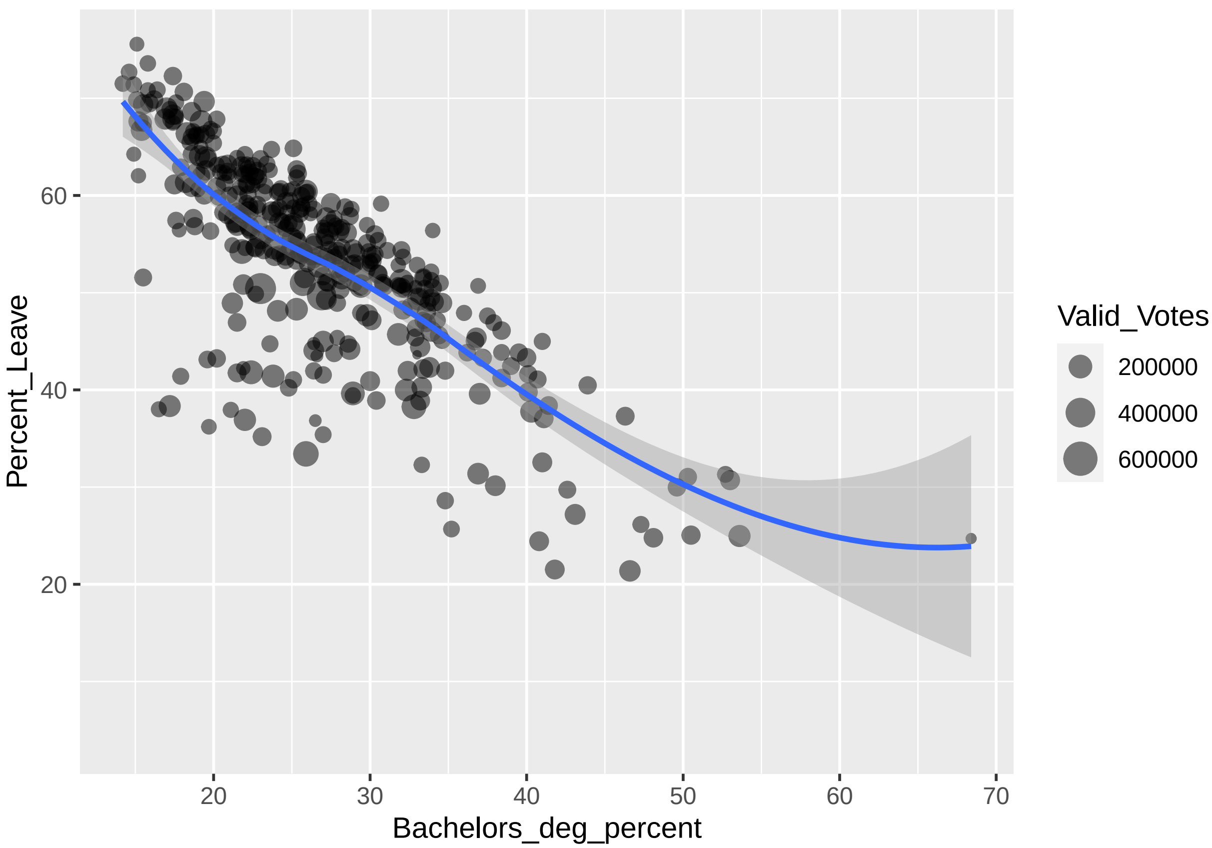
3f) Make a figure showing `Bachelors_deg_percent` on the horizontal axis and `Percent_Leave` on the vertical axis. Include a dot for each local authority, with the size scaled by `Valid_Votes` and specifying `alpha = .5` in your `geom_point()` command to avoid excessive overplotting. Use `geom_smooth()` to estimate the CEF. Does the relationship look linear?

```
brexit |>
  ggplot(aes(x = Bachelors_deg_percent, y = Percent_Leave)) +
  geom_point(aes(size = Valid_Votes), alpha = .5) + # +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 7 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```



It looks quite linear, yes! The `geom_smooth()` curves up on the right, but there is really only one data point there.

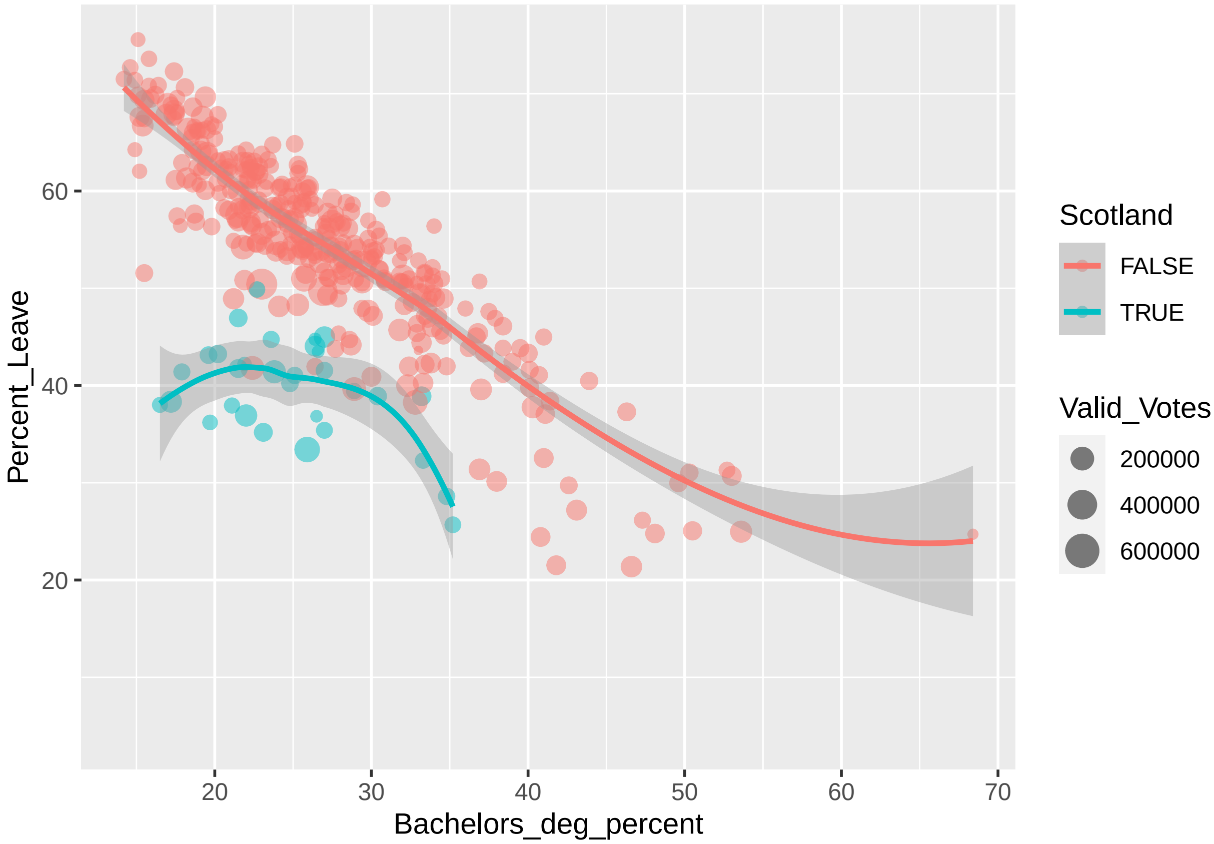
3g) Do the same, but estimate the CEF separately for Scotland and the rest of the sample. (Hint: create a variable that distinguishes Scotland from other places, and assign it to the `color` aesthetic.) Describe the result in words.

```
brexit |>
  mutate(Scotland = Region == "Scotland") |>
  ggplot(aes(x = Bachelors_deg_percent, y = Percent_Leave, color = Scotland)) +
  geom_point(aes(size = Valid_Votes), alpha = .5) + # +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 7 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```



It appears that the plotted relationship is different in Scotland than in the rest of the UK. Support for Brexit is lower at each level of education in Scotland than elsewhere, and support for Brexit also seems less strongly related to education in Scotland than elsewhere.

3h) Based on what you found in (3g), run a regression predicting support for Leave in a local authority as a function of the proportion of residents with a bachelors degree and whether the local authority is in Scotland. (Do you include an interaction? Explain why or why not.) Report the result in a regression table as above. According to your model, what is the predicted support for Brexit in a Scottish local authority in which 30% of inhabitants have a bachelor's degree?

```
brexit |>
  mutate(Scotland = Region == "Scotland") -> brexit2

lm(Percent_Leave~ Bachelors_deg_percent*Scotland, data = brexit2) |>
  huxtable::huxreg()
```

Based on (3g), I think it's appropriate to model support for Leave as a function of the proportion of inhabitants with bachelor's degrees, a dummy for Scotland, and the interaction.

In this model, predicted support for Leave in a Scottish local authority in which 30% of inhabitants have a bachelor's degree is $84.575 + 30 \times -1.113 - 32.711 + 30 \times .618 = 37.014$.

	(1)
(Intercept)	84.576 ***
	(0.879)
Bachelors_deg_percent	-1.113 ***
	(0.031)
ScotlandTRUE	-32.711 ***
	(4.318)
Bachelors_deg_percent:ScotlandTRUE	0.618 ***
	(0.170)
N	375
R2	0.811
logLik	-1096.363
AIC	2202.727

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.