Практикум 1. Введение в работу с данными в R. Коэффициент Пирсона и простая линейная регрессия.

Алла Тамбовцева

Содержание

Введение: контекст исследования и описание данных

В рамках этого практикума мы будем работать с данными, которые использовались в исследовании P.Loewen et al. "A Natural Experiment in Proposal Power and Electoral Success", посвященном факторам, влияющим на процент голосов за кандидатов, избираемых в Палату общин Канады. В числе прочего, авторы задавались следующими вопросами:

- правда ли, что кандидаты, которые обладают возможностью вносить предложения[1] по бюджету и другим вопросам, в целом, получают больше голосов на выборах;
- отличается ли характер взаимосвязи между процентом голосов на выборах и реальной возможностью вносить предложения во время слушаний у кандидатов, которые занимают пост в правительстве, и кандидатов, которые с правительством не аффилированы.

Полностью воспроизвести регрессионные модели, которые строят авторы, мы пока не сможем, но освежить в памяти графики и статистические тесты для сравнения групп – вполне! К тому же мы проанализируем взаимосвязь между процентом голосов на текущих выборах и на предыдущих – связь, которая часто оценивается в электоральных исследованиях и которую не обошли вниманием авторы статьи.

В рамках практикума мы будем работать с сокращенной версией набора данных, сохраненной в CSV-файле. Полную версию (формат STATA) можно найти на Harvard Dataverse.

[1] С 2004 года для членов Палаты устраивается лотерея, в ходе которой между ними распределяются места в списке очередности, который определяет порядок, когда члены парламента могут выдвигать свои предложения и выносить их на голосование. Само по себе наличие такой лотереи позволяет наблюдать за естественным экспериментом. Эксперимент — потому что места распределяются случайно, естественный — потому что условия не контролируются исследователями, а являются реально существующей формальной процедурой. На основе мест в списке очередности авторы создают бинарный индикатор: 0 – no proposal power, 1 – proposal power.

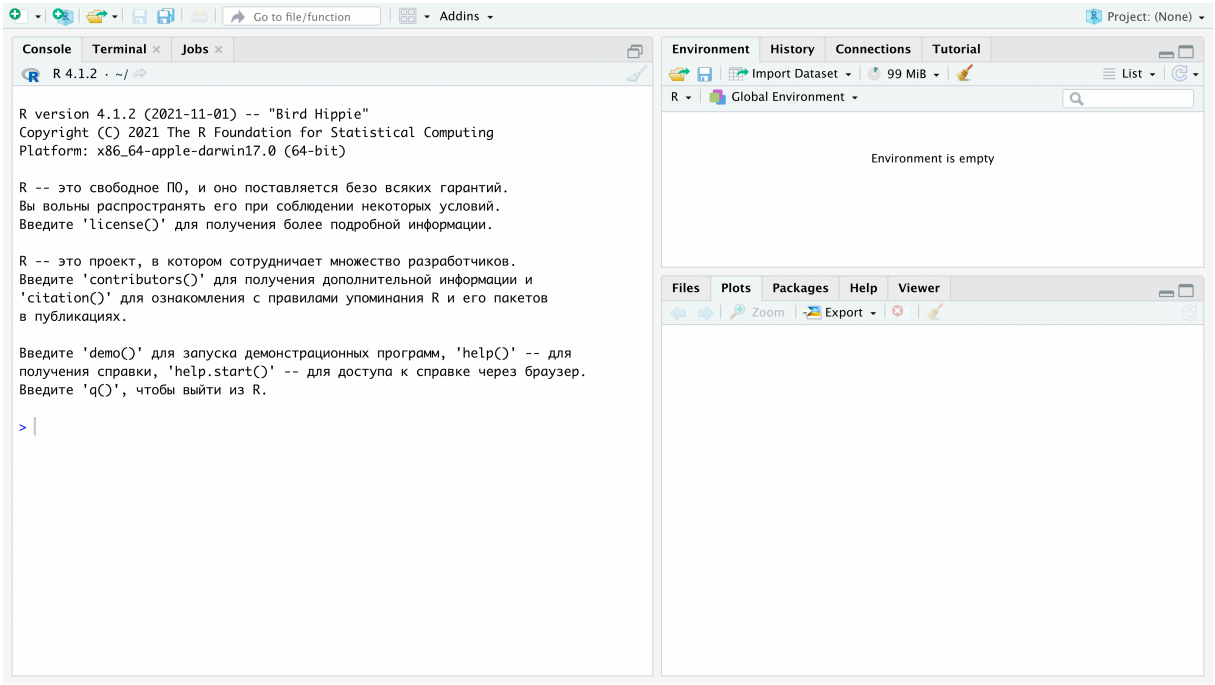
Переменные в файле:

- `election`: год выборов;
- `province` и `district`: регион и избирательный округ;
- `candidate`: имя кандидата;
- `media_mentions`: число упоминаний в СМИ;
- `current_vote` и `previous_vote`: процент голосов на текущих и предыдущих выборах;
- `liberal-other`: относится ли кандидат к соответствующей партии;
- `gender`: пол кандидата;
- `place`: место в списке очередности;
- `p2p`: обладает ли кандидат возможностью вносить предложения во время слушаний;
- `government`: занимает ли кандидат пост в правительстве.

Загрузка файлов и запуск кода

Загрузка файлов в RStudio

1. Скачиваем файл с кодом `regression-practice01.R` и файл с данными `canada.csv`.
2. Запускаем RStudio. Интерфейс RStudio выглядит так:

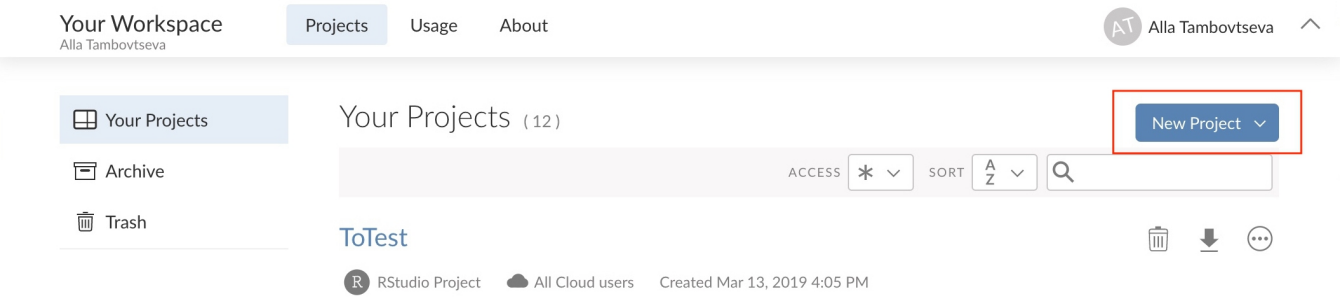


   Если интерфейс сильно отличается, проверьте два момента: что вы запустили RStudio, а не R; что вы установили и открыли среду RStudio, а не программу R-Studio для дефрагментации жесткого диска.

3. В меню RStudio выбираем *File* → *Open*. В открывшемся окне (если не появилось, проверьте, оно иногда скрывается на панели управления внизу, где меню *Пуск* или значки программ) выбираем файл `regression-practice01.R`, кликаем *Open*.

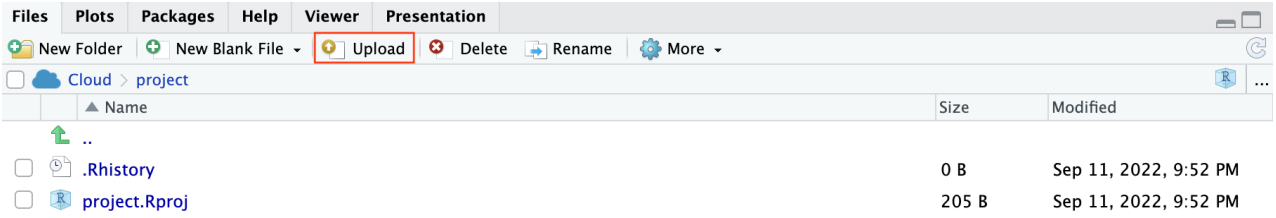
Загрузка файлов в RStudio Cloud

1. Скачиваем файл с кодом `regression-practice01`, скачиваем файл с данными `canada.csv`.
2. Заходим в RStudio Cloud, залогиниваемся, создаем новый проект через *New Project* → *New RStudio Project*:



   Можно заходить в уже существующий проект, в нем может быть сколько угодно файлов, один проект – это одна рабочая область, примерно как папка, только в облаке.

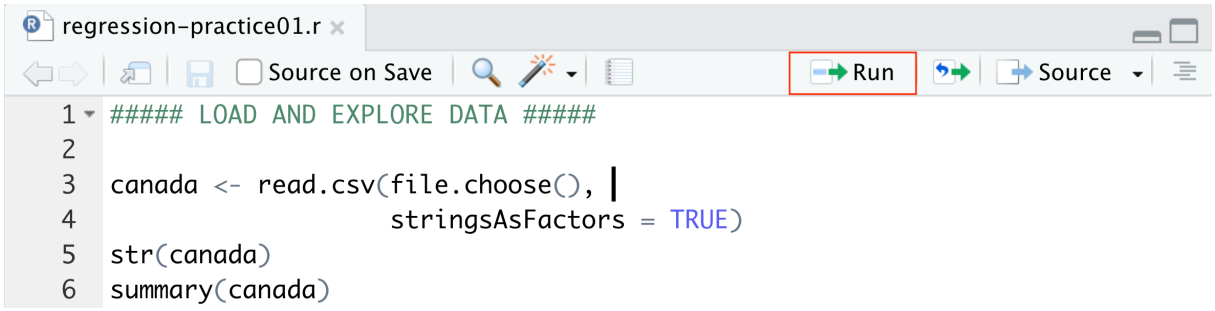
3. В правом нижнем углу во вкладке `Files` кликаем на кнопку `Upload`, загружаем файл `regression-practice01.R`. Потом проделываем ту же операцию с `canada.csv`.



4. Кликаем в `Files` на название файла с кодом R, он открывается.

Запуск кода

- Код обычно запускаем построчно: ставим курсор на нужную строчку (в любом месте, необязательно в конце строки) и либо нажимаем *Ctrl* + *Enter* (*Command* + *Enter*), либо кликаем на кнопку *Run*:



- Можно выделить сразу несколько строк и запустить аналогичным образом.
- Если при запуске какой-то строки в консоли выводится ошибка вида `object not found`, проверьте, а была ли строка с созданием этого объекта запущена ранее. R не знает, в каком месте файла с кодом находится та или иная строка, для него важна именно последовательность запуска. Запущена строка кода или нет, можно проверить в консоли.

Загрузка данных и знакомство с переменными

Важно: после скачивания CSV-файла не открывайте его в Excel/Numbers и иных программах, так как это обычно приводит к изменению его изначального формата, и к тому, что в R он будет загружен некорректно.

Загрузим данные, которые хранятся в файле `canada.csv`:

```
canada <- read.csv(file.choose(), stringsAsFactors = TRUE)
```

Пояснения к коду:

- функция `read.csv()` загружает в R данные из CSV-файла;
- функция `file.choose()` открывает окно для выбора файла на компьютере (используем, чтобы не писать путь к файлу и не искать рабочую папку);
- опция `stringsAsFactors = TRUE` нужна для более удобной работы с текстовыми столбцами — каждому уникальному текстовому значению присваивается понятная R числовая метка.

После загрузки во вкладке *Environment* появится объект `canada`, это будет датафрейм с 404 строками и 19 столбцами. Можем кликнуть на название `canada` в *Environment*, тогда датафрейм откроется в отдельной вкладке.

Выведем структуру датафрейма – техническое описание всех столбцов (для компактности выдача урезана):

```
str(canada)
```

```
## 'data.frame':    404 obs. of  7 variables:
##  $ media_mentions: int  44 60 66 330 1 9 113 75 52 268 ...
##  $ current_vote  : num  46.5 57.1 50.2 46.9 46 ...
##  $ previous_vote : num  57.1 59.8 51.7 41.5 47.9 ...
##  $ gender        : Factor w/ 2 levels "Female","Male": 2 2 1 1 2 2 2 2 2 2 ...
##  $ year_served   : int  8 6 20 9 4 2 15 2 13 2 ...
##  $ p2p           : int  0 0 0 1 1 0 1 0 0 1 ...
##  $ government    : int  0 1 0 0 0 0 0 0 1 1 ...
```

Расшифровка типов данных:

- `int` – целочисленный тип, от *integer*;
- `num` – числовой тип, от *numeric*;
- `factor` – факторный тип, то есть текст с присвоенными числовыми метками.

Теперь выведем описательные статистики (для компактности выдача урезана):

```
summary(canada)
```

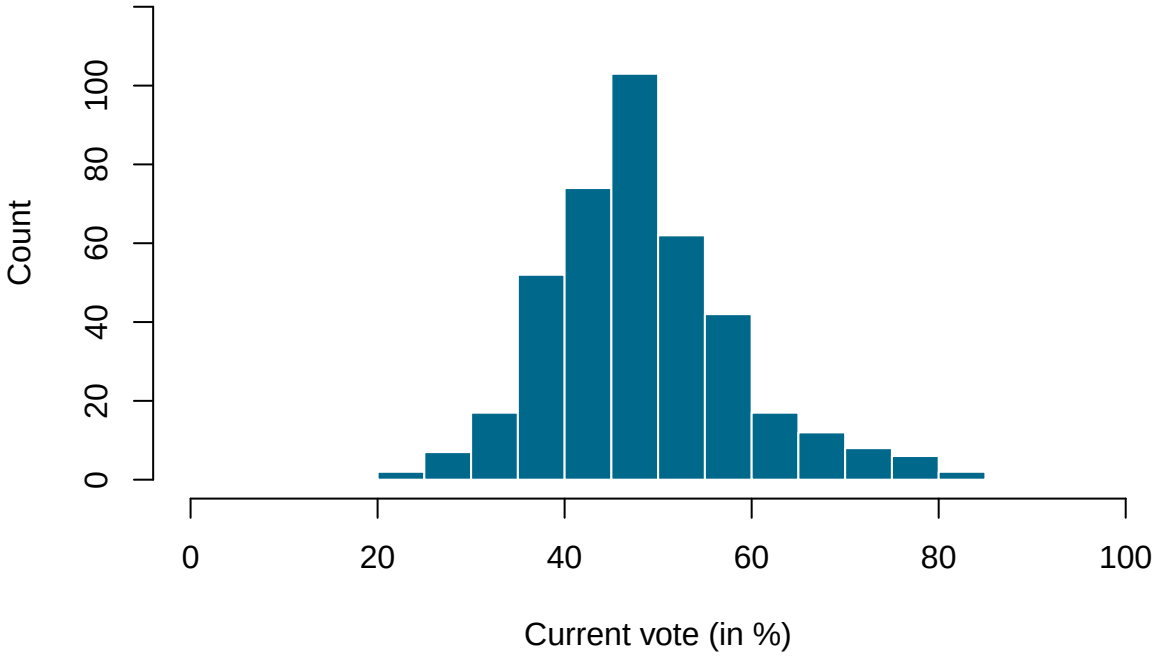
```
##  media_mentions      current_vote   previous_vote      gender
##  Min.   :    0.00   Min.   :24.19   Min.   :26.75   Female: 79
##  1st Qu.:   28.25   1st Qu.:41.50   1st Qu.:42.28   Male  :325
##  Median :   88.50   Median :47.90   Median :47.94
##  Mean   :  415.73   Mean   :48.39   Mean   :49.11
##  3rd Qu.:  285.50   3rd Qu.:53.98   3rd Qu.:55.06
##  Max.   :22615.00   Max.   :82.56   Max.   :82.56
##  NA's   :2
##   year_served          p2p           government
##  Min.   : 2.000   Min.   :0.0000   Min.   :0.000
##  1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:0.000
##  Median : 6.000   Median :0.0000   Median :0.000
##  Mean   : 6.958   Mean   :0.4084   Mean   :0.255
##  3rd Qu.:11.000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :27.000   Max.   :1.0000   Max.   :1.000
##
```

Для числовых столбцов выводится минимум, максимум, среднее арифметическое, медиана, нижний квартиль и верхний квартиль. Для факторных – частоты для уникальных значений.

Изучение распределения данных

Построим гистограмму, которая покажет нам распределение процентов голосов за кандидатов на текущих выборах:

```
hist(canada$current_vote,
     col = "deepskyblue4",
     border = "white",
     main = "",
     xlab = "Current vote (in %)",
     ylab = "Count",
     xlim = c(0, 100),
     ylim = c(0, 120))
```



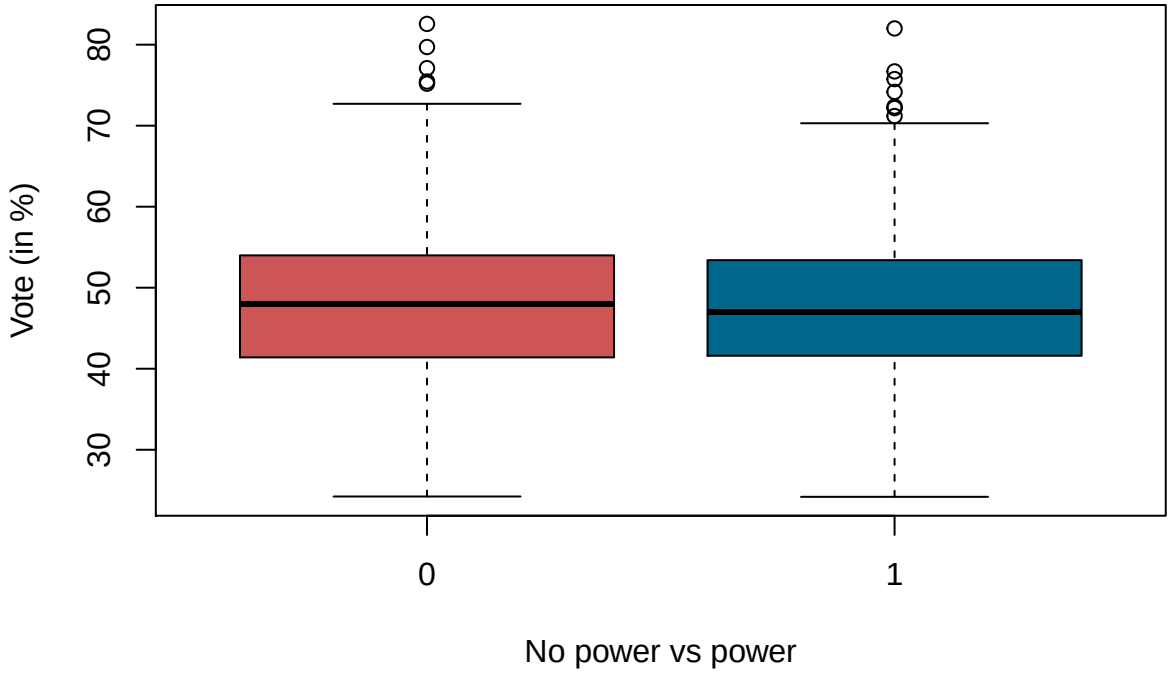
Видно, что распределение почти симметричное, однако есть небольшая скошенность вправо – есть кандидаты, за которых проголосовало около 80% избирателей. Такой процент довольно высок относительно остальных (большинство значений сконцентрировано на отрезке 40-60%).

Пояснения к коду:

- функция `hist()` строит гистограмму для числового вектора или столбца таблицы;
- аргумент `col` задает цвет заливки, если его не указать, гистограмма по умолчанию будет серой;
- аргумент `border` задает цвет границ столбцов, если его не указать, по умолчанию будет выбран черный цвет;
- аргумент `main` задает заголовок графика, здесь он пустой, по умолчанию будет надпись вида `Histogram of canada$current_vote`, выровненная по центру;
- аргументы `xlab` и `ylab` задают подписи по горизонтальной и вертикальной осям;
- аргументы `xlim` и `ylim` задают границы осей (минимальное и максимальное значение); можно не выставлять, R подберет их сам, но иногда полезно отрегулировать масштаб, например, для более удобных сравнений нескольких графиков.

Теперь посмотрим, насколько различается распределение процента голосов за кандидатов, обладающих властью выдвигать предложения и тех, кто такой властью не обладает. Построим ящики с усами для двух групп:

```
boxplot(canada$current_vote ~ canada$p2p,
        col = c("indianred3", "deepskyblue4"),
        xlab = "No power vs power",
        ylab = "Vote (in %)")
```



Особой разницы в распределениях не видно.

Пояснения к коду:

- оператор $\sim$ означает зависимость; в данном случае мы строим график для `current_vote` в зависимости от того, какое значение в `p2p`, то есть с учетом деления на группы;
- здесь два цвета заливки, поэтому они оформлены в виде вектора, помещены внутрь `c()`.

Проверим более формально, если ли разница в средних процентах голосов за кандидатов, имеющих возможность вносить предложения и не имеющих такой возможности. Применим критерий Стьюдента для двух выборок[2]:

```
t.test(canada$current_vote ~ canada$p2p)
```

```
##
##  Welch Two Sample t-test
##
## data:  canada$current_vote by canada$p2p
## t = -0.85847, df = 332.66, p-value = 0.3913
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.920805  1.146007
## sample estimates:
## mean in group 0 mean in group 1
##        48.02515        48.91255
```

[2]Тест, используемый в R, не предполагает равенства дисперсий генеральных совокупностей, он же тест Уэлча. Если действовать совсем строго и консервативно, перед выбором теста для сравнения групп нужно проверить распределение групп на нормальность, но при больших выборках на практике этим часто пренебрегают.

Итак, нулевая гипотеза и альтернативная гипотеза:

$$H_0 : \mu_{\text{no power}} = \mu_{\text{power}}$$

$$H_1 : \mu_{\text{no power}} \neq \mu_{\text{power}}$$

Если принять уровень значимости равным 5%, то нулевую гипотезу не следует отвергать (p-value = 0.3913). Действительно, средний процент голосов в двух группах можно считать одинаковым. Разница в выборочных средних составляет примерно 1 процентный пункт, но ее недостаточно для того, чтобы считать различия заметными.

Все-таки получать незначимый результат обидно. Поэтому посмотрим на вопрос иначе: что если разница в результатах есть, но она проявляется не для всех кандидатов, а только для тех, которые формально имеют больше полномочий? Другими словами, давайте сравним те же группы, но предварительно разделим кандидатов на тех, кто занимает пост в правительстве и тех, кто не занимает.

Отфильтруем строки датафрейма и сохраним их в отдельные датафреймы `government` и `opposition`:

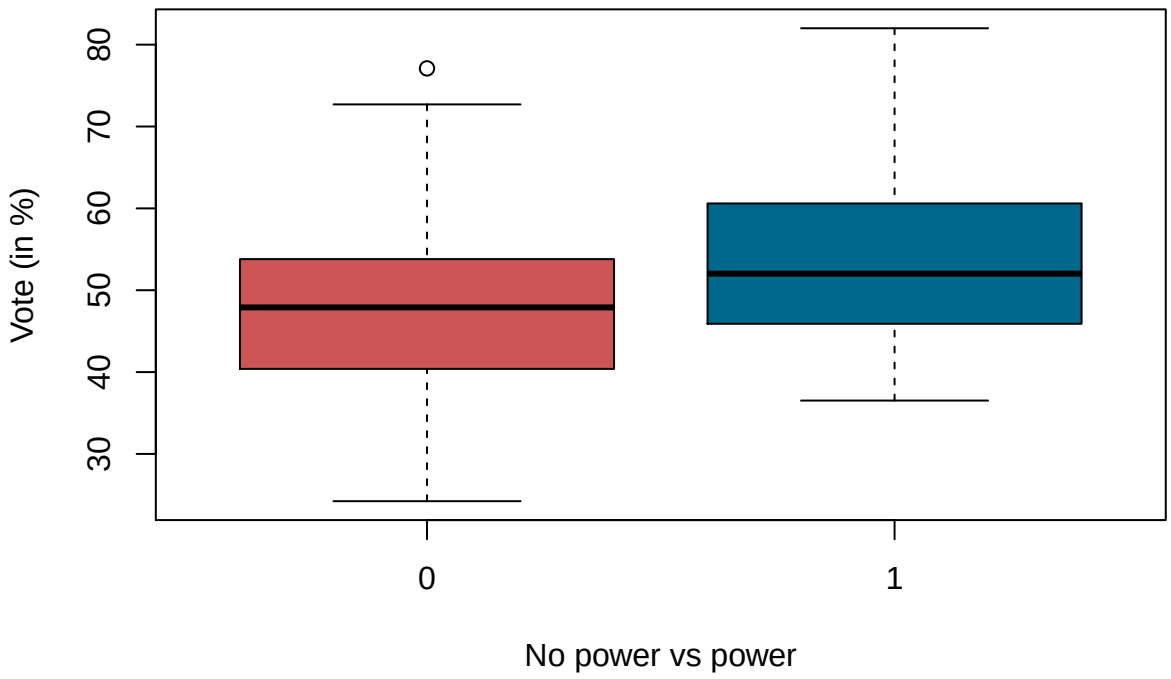
```
government <- canada[canada$government == 1, ]
opposition <- canada[canada$government == 0, ]
```

Пояснения к коду:

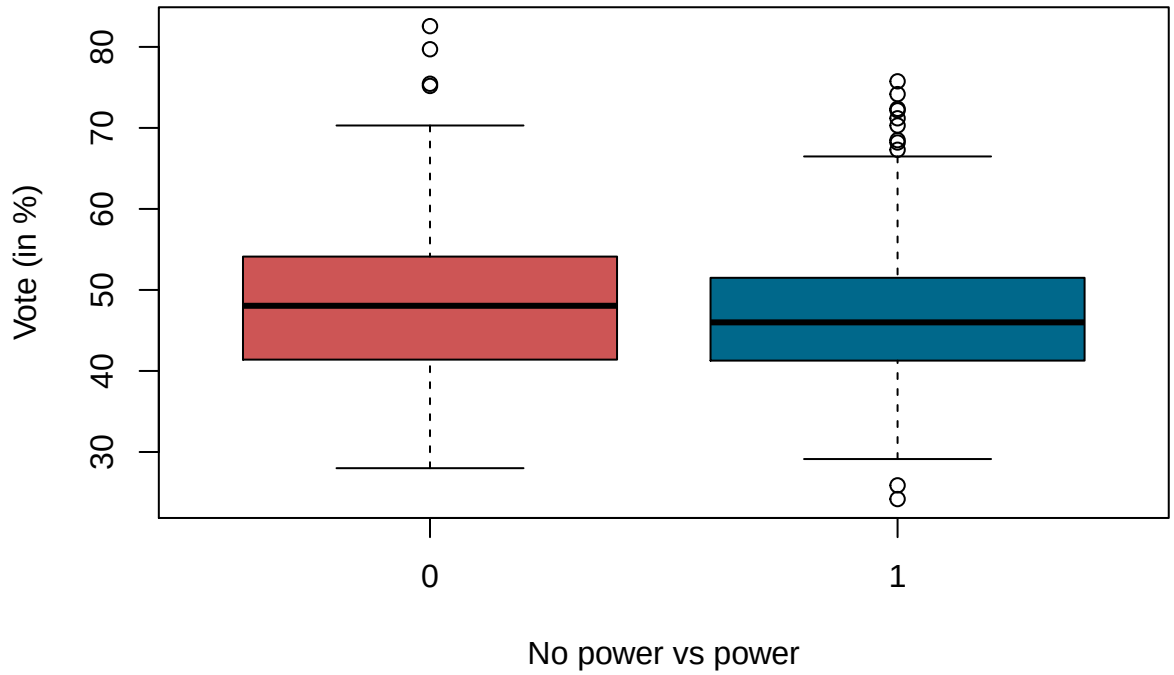
- квадратные скобки используются для фильтрации, до запятой указывается условие на строки, после – на столбцы; здесь нет условия на столбцы, выбираем все, что есть, поэтому после запятой пусто;
- для проверки условия равенства в программировании используется оператор `==`; выражение в квадратных скобках возвращает значения `TRUE` или `FALSE`, и R отбирает из `canada` только те строки, на которых было возвращено `TRUE`.

Построим аналогичные ящики с усами для двух групп, отдельно для кандидатов, аффилированных с правительством, и не аффилированных с правительством:

```
boxplot(government$current_vote ~ government$p2p,
        col = c("indianred3", "deepskyblue4"),
        xlab = "No power vs power",
        ylab = "Vote (in %)")
```



```
boxplot(opposition$current_vote ~ opposition$p2p,
        col = c("indianred3", "deepskyblue4"),
        xlab = "No power vs power",
        ylab = "Vote (in %)")
```



Здесь уже все более интересно. Если кандидаты занимают посты в правительстве и имеют возможность выдвигать предложения на слушаниях, они получают заметно больше голосов. В случае, если кандидаты с правительством не связаны, возможность выдвигать предложения существенным образом на проценте голосов не отражается.

Для надежности посмотрим на результаты формальных тестов:

```
t.test(government$current_vote ~ government$p2p)
```

```
##
##  Welch Two Sample t-test
##
## data:  government$current_vote by government$p2p
## t = -2.6028, df = 75.094, p-value = 0.01114
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -9.749175 -1.295838
## sample estimates:
## mean in group 0 mean in group 1
##        47.67723        53.19974
```

```
t.test(opposition$current_vote ~ opposition$p2p)
```

```
##
##  Welch Two Sample t-test
##
## data:  opposition$current_vote by opposition$p2p
## t = 0.45167, df = 259.94, p-value = 0.6519
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.765004  2.815706
```

```
## sample estimates:
## mean in group 0 mean in group 1
##        48.15511        47.62976
```

Наши предположения, сделанные на основе графиков подтвердились. В первом случае различия в средних значениях есть (p-value = 0.011), во втором – нет (p-value = 0.652).

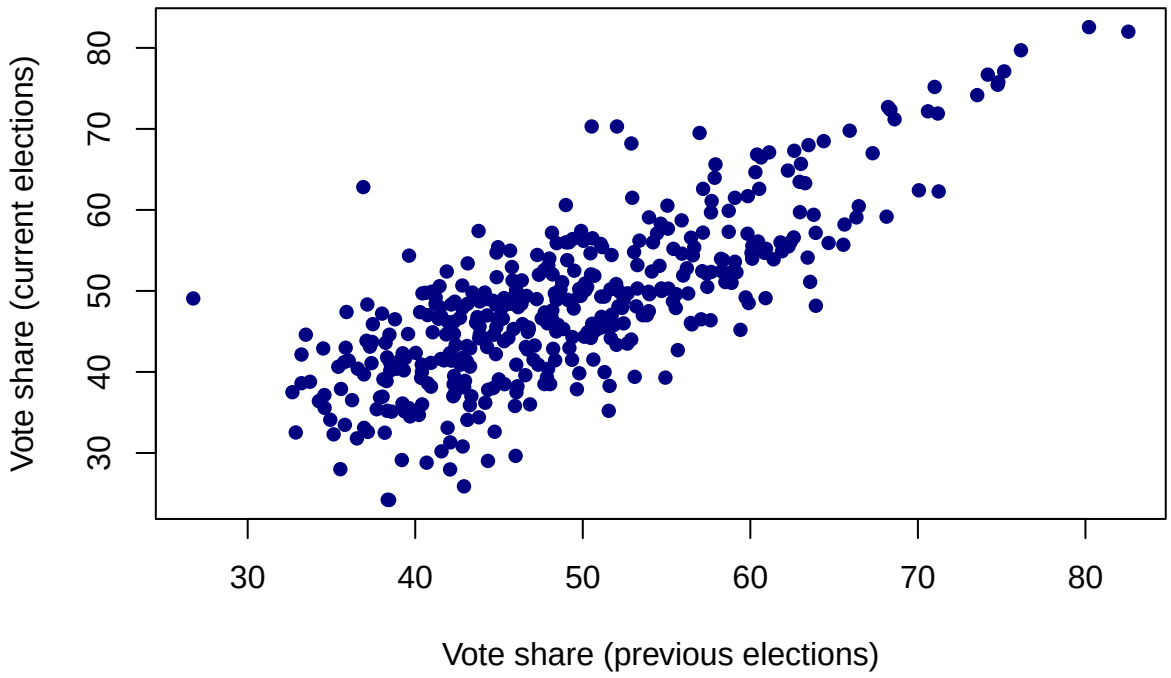
Учитывая, что наш курс все-таки про регрессионный анализ, было бы логично завершить это мини-исследование моделью, которая показала бы, как процент голосов за кандидатов зависит от наличия возможности влиять на принятие решений и от аффиляции с правительством. Такая модель возможна, но пока у нас недостаточно знаний, чтобы ее правильно проинтерпретировать. Поэтому перейдем к вопросам попроще.

Анализ взаимосвязей

Диаграмма рассеивания и коэффициент Пирсона

Изучим связь между процентом голосов на текущих выборах и предыдущих. Для начала построим диаграмму рассеивания:

```
plot(canada$previous_vote, canada$current_vote,
     pch = 16, col = "navy",
     xlab = "Vote share (previous elections)",
     ylab = "Vote share (current elections)")
```



Видно, что связь между показателями прямая, сильная или близкая к сильной.

Пояснения к коду:

- функция `plot()` строит обычный точечный график, внутри мы указываем, что идет по горизонтальной и вертикальной оси;
- аргумент `pch` определяет тип точки, все типы можно посмотреть в документации, запустив строку кода `?pch`.

Проверим наличие линейной связи более формально – с помощью коэффициента корреляции Пирсона и связанного с ним статистического критерия:

```r
cor.test(canada$previous_vote, canada$current_vote)
```

```
##
##  Pearson's product-moment correlation
##
## data:  canada$previous_vote and canada$current_vote
## t = 24.628, df = 402, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7334313 0.8116565
## sample estimates:
##       cor
## 0.7755032
```

Итак, значение коэффициента довольно высокое, 0.775, связь близка к сильной. Значение p-value примерно 0 (меньше, чем 2.2×10^{-16}), значит, нулевая гипотеза о равенстве истинного коэффициента корреляции нулю отвергается, что говорит нам о наличии линейной связи между показателями.

Парная линейная регрессия

Теперь изучим эту связь более детально. Построим модель парной линейной регрессии, которая покажет нам, как именно процент голосов на текущих выборах зависит от процента голосов на предыдущих выборах.

Стоит отметить, что в данном случае речь не идет о причинно-следственной связи, регрессия, как и корреляция, ее не выявляет. Однако мы точно знаем направленность связи: процент голосов на предыдущих выборах может влиять на процент голосов на текущих выборах, никак не наоборот, машину времени еще не изобрели.

Уравнение модели можно записать так:

$$\hat{\text{current\_vote}} = \hat{b_0} + \hat{b_1} \times \text{previous\_vote}$$

где $\hat{b_0}$ и $\hat{b_1}$ – оценки коэффициентов, которые посчитает R, используя метод наименьших квадратов, а «крышечка» над `current_vote` означает, что это значения процента голосов, предсказанные нашей моделью, а не настоящие, которые мы видим в столбце таблицы с данными.

Как мы сможем проинтерпретировать полученные коэффициенты $\hat{b_0}$ и $\hat{b_1}$?

- Коэффициент $\hat{b_0}$: среднее значение `current_vote`, если `previous_vote` равно 0. Это то значение, где регрессионная прямая пересекает вертикальную ось.
- Коэффициент $\hat{b_1}$: показывает, на сколько, в среднем, изменяется `current_vote`, если `previous_vote` увеличивается на 1.

Запустим модель линейной регрессии в R. Для построения линейных моделей используется функция `lm()`, что расшифровывается как *linear model*. Уравнение имеет вид `y ~ x`, где перед ~ указывается зависимая переменная, а после ~– независимая (вообще независимых может быть несколько, но пока мы говорим о парных моделях):

```r
lm(data = canada, current_vote ~ previous_vote)
```

```
##
## Call:
## lm(formula = current_vote ~ previous_vote, data = canada)
##
## Coefficients:
##   (Intercept)  previous_vote
##        7.5348         0.8319
```

Итак, R выдал нам оценки коэффициентов модели. Здесь `Intercept` – это $\hat{b_0}$, а `previous_vote` — коэффициент при соответствующей переменной, то есть $\hat{b_1}$. Теперь можем записать уравнение регрессии с конкретными числами:

$$\hat{\text{current}}\_\text{vote} = 7.53 + 0.83 \times \text{previous}\_\text{vote}$$

Проинтерпретируем его:

- если на предыдущих выборах кандидат набрал 0% голосов, то на текущих выборах, в среднем, стоит ожидать 7.83% (не совсем реалистичная ситуация, но формально это так; из-за некоторой нереалистичности этот коэффициент в социально-экономических исследованиях часто не интерпретируют);
- при увеличении процента голосов на предыдущих выборах на 1 процентный пункт, процент голосов на текущих выборах, в среднем, увеличивается на 0.83 процентных пункта (прошлое изменять мы не можем и процент на прошлых выборах тоже, но в данном случае можно перевести это в некоторый прогноз: если на текущих выборах кандидату удастся увеличить процент голосов на 1 процентый пункт, то на следующих, в среднем, согласно нашей модели, можно ожидать рост процента голосов на 0.83 процентных пункта).

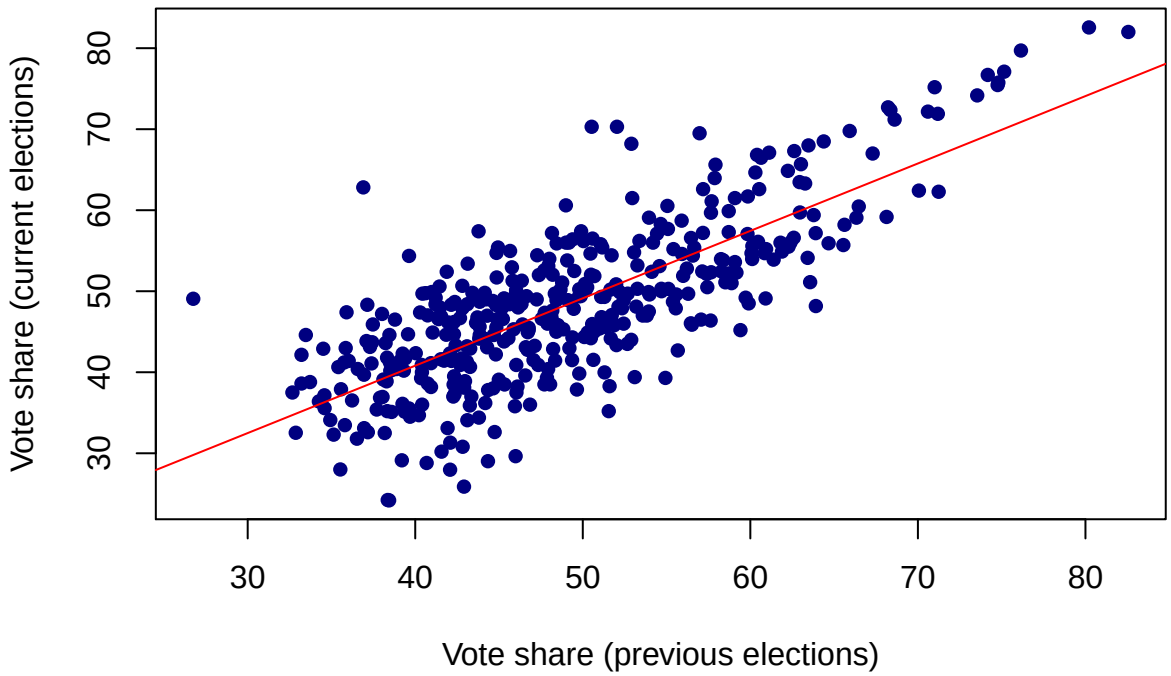
Получив уравнение выше, мы получили возможность делать прогнозы. Если нам известен процент голосов за кандидата на прошлых выборах, но не известен на текущих, мы сможем его предсказать с помощью модели. Посчитаем ожидаемый процент голосов за кандидата, если известно, что на прошлых выборах он набрал 60%:

$$\hat{\text{current}}\_\text{vote} = 7.53 + 0.83 \times 60 = 57.33$$

Также мы можем наложить регрессионную прямую на диаграмму рассеивания:

```
plot(canada$previous_vote, canada$current_vote,
     pch = 16, col = "navy",
     xlab = "Vote share (previous elections)",
     ylab = "Vote share (current elections)")

abline(a = 7.5348, b = 0.8319, col = "red")
```



Пояснения к коду:

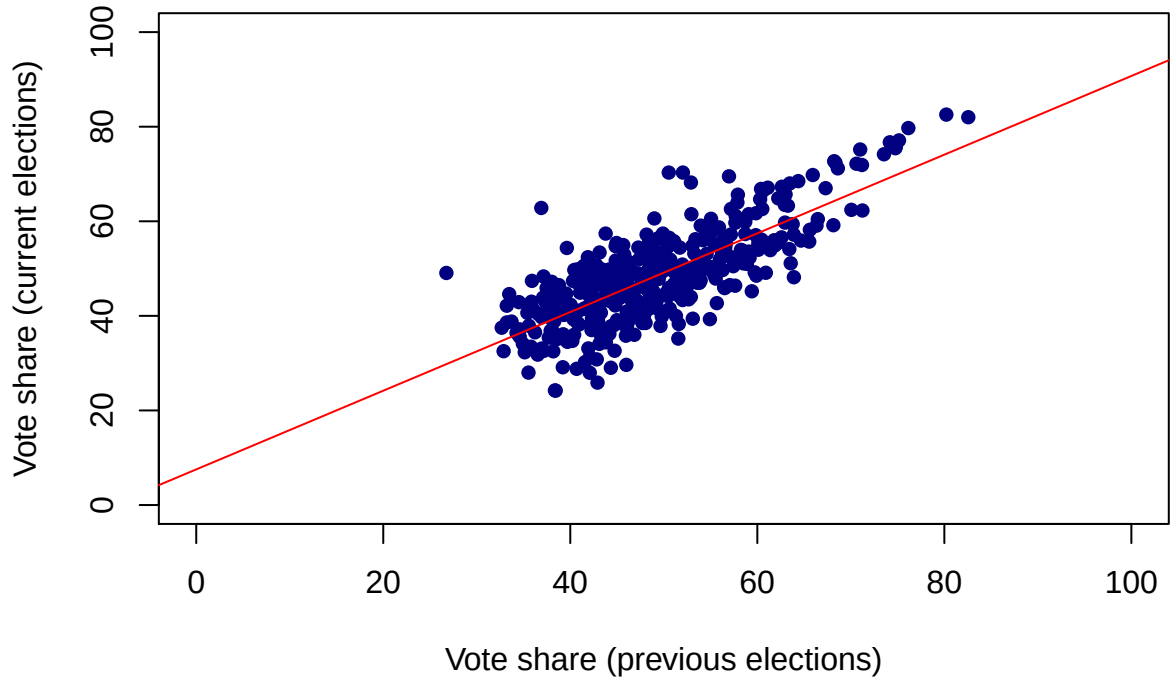
- функция `abline()` добавляет прямую на график, построенный ранее;

- прямая имеет вид `a + bx`, поэтому на место `a` мы поставили $\hat{b_0}$, а на место `b` – значение $\hat{b_1}$.

Логичный вопрос: почему здесь не видно, что прямая пересекает вертикальную ось в значении 7.53? Почему прямая упирается в точку около 30? Все просто: график немного обрезан, мы не видим начало координат. Можем изменить масштаб, и тогда увидим все, как надо:

```
plot(canada$previous_vote, canada$current_vote,
     pch = 16, col = "navy",
     xlab = "Vote share (previous elections)",
     ylab = "Vote share (current elections)",
     xlim = c(0, 100),
     ylim = c(0, 100))

abline(a = 7.5348, b = 0.8319, col = "red")
```



Напоследок посмотрим, что еще хранится внутри модели. На лекции обсуждалась логика метода наименьших квадратов (МНК). Идея метода несложная: нужно подобрать такие значения $\hat{b_0}$ и $\hat{b_1}$, чтобы общая ошибка модели была минимальной. Общая ошибка модели определяется как сумма квадратов ошибок, то есть сумма квадратов разностей настоящих значений зависимой переменной (y) и предсказанных моделью ($\hat{y}$):

$$\sum_i (y_i - \hat{y_i})^2$$

Геометрически это означает, что на диаграмме рассеивания через облако точек мы должны провести прямую таким образом, чтобы сумма квадратов расстояний от точек до прямой была минимальной из всех возможных (но при этом она совсем не обязана быть 0).

Так вот: предсказанные значения зависимой переменной (`fitted.values`) и ошибки модели (`residuals`) для каждого наблюдения можно извлечь из R в явном виде!

Сохраним модель в переменную `m1`:

```
m1 <- lm(data = canada, current_vote ~ previous_vote)
```

Запросим для первых шести кандидатов значения процента голосов, предсказанные нашей моделью (функция `head()` выдает первые шесть значений, для компактности не будем выводить все 404 наблюдения):

```
head(m1$fitted.values)
```

```
##        1        2        3        4        5        6
## 55.00065 57.30490 50.50029 42.09851 47.37250 53.74454
```

А теперь ошибки модели для этих кандидатов:

```
head(m1$residuals)
```

```
##          1          2          3          4          5          6
## -8.5006502 -0.2448992 -0.3002912  4.7814890 -1.3724996 -5.8545416
```

Обратите внимание: где-то наша модель ошибается «в плюс», а где-то «в минус». У каких-то наблюдений ошибки отрицательные (модель завышает процент голосов), а у каких-то — положительные (модель занижает процент голосов).

Можем посмотреть на квадраты ошибок:

```
head(m1$residuals ^ 2)
```

```
##           1           2           3           4           5           6
## 72.26105383  0.05997563  0.09017483 22.86263686  1.88375516 34.27565767
```

И даже их суммировать:

```
sum(m1$residuals ^ 2)
```

```
## [1] 16225.55
```

Полученное число кажется ужасающе большим. Но это нестрашно: во-первых, оно зависит от числа наблюдений, а оно здесь немаленькое; во-вторых, оно зависит от единиц измерения и общего разброса значений. В следующем практикуме мы обязательно выясним, как определить, насколько хороша предсказательная сила модели и насколько часто она ошибается.