Example Exploratory Data Analysis

This is an unmarked optional tutorial to show the kind of thinking that goes into an exploratory data analysis

The goal of this tutorial document is to walk through some of the common issues encountered in the early stages of an exploratory analysis on a set of data. It gives examples of common problem areas in:

* reading in data
* dealing with blanks
* dealing with factors

This data is a modified version of data from the New Zealand Election Survey, deliberately modified to introduce problems that occur naturally in many data sets.

## Step One. Learn something about the data set.

In this case, the New Zealand Election Survey takes place every three years as a postal survey of a sample of registered electors. Some sampled electors were part of a sample panel of people surveyed at the previous election as part of a longitudinal study, others were randomly chosen from the electoral roll. Those electors that were part of the longitudinal panel group were randomly selected in previous elections.

As well as survey results, the data set includes information from the electoral roll, and weighting values for adjusting results. The full NZES data set has been reduced to a selected group of variables, making 3101 observations of 107 variables.

## Step Two. Contemplate some questions.

Examining the codebook (or in this case the appendix at the end of the document to check out the variables).

For example, we might decide that since New Zealand is a Mixed Member Proportional voting system, where people get to vote for both an electorate (local) representative and a nationwide political party, that it would be interesting to look at strategic voting under conditions where there are many political parties to choose from. We identify some relevant variables of interest in the data, and investigate the nature of the individual variables before we explore their interactions. The kind of variables they are is going to shape our question.

### Read in the data

There are many different kinds of data files in the world. Each one has its own issues when being read in by R. In this case the data is saved as a .RData file, which can be read in by using the load() command.

As with most reading in file commands, inside the parentheses needs to go a piece of text, in quotes, that is the path to the file from the working directory (the working directory is the folder that R is currently paying attention to). The easiest way to get a R Markdown (Rmd) Document and console cooperating about this is to place the file with the data in it in the same folder as the R Markdown (Rmd) Document, open the R Markdown (Rmd) document in RStudio so we are looking at the contents of the document in the editing window, then in the RStudio Session menu, use the **Set Working Directory - To Source File Location** command to make a common starting point. Then the code in the R Markdown (Rmd) document will use the same working space regardless of whether we knit the document or run code chunks in the Console. In this case, if the nzes2011.RData file is in the same folder as the R Markdown (Rmd) document, hence it can be read into R with the following command:

load("selected\_nzes2011.Rdata")

We also want to load packages that have functions in them we want to use. For this particular analysis we will only need the dplyr package, but for your project you will also likely need other packages as well, e.g. ggplot2.

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.0.5

## Step Three. Prepare for the first question

As a first question, we might be interested in exploring the relationship between the party the person voted for, the party that was their favourite, and if they believed that their vote makes a difference -- focusing on the question that are people who believe their vote makes a difference more likely to strategically vote for a party not their favourite. To achieve this, we familiarise ourselves with the variables jpartyvote, jdiffvoting, and \_singlefav. First we check the codebook (see Appendix), then we explore the data.

Viewing the entire dataset in the Data Viewer window by clicking on the data frame's name in the Environment or running the View() command in the Console can be ineffective since the Data Viewer only shows the first 100 columns of the data frame.

Using the str() command on the entire dataset can also be equally ineffective. However we can subset the columns of interest and take a closer look at them. We can use the dplyr chain to select the variables of interest and investigate only their structure by adding str() at the end of the chain:

selected\_nzes2011 %>%   
 select(jpartyvote, jdiffvoting, \_singlefav) %>%   
 str()

If we try to run that line, we will get an error message about unexpected input or missing object.

We next need to diagnose where the problem lies -- in the R code or in the data? The best way to troubleshoot this issue is to run each line of the dplyr chain one by one.

selected\_nzes2011

The first line runs without any erros, but the second line gives an error

selected\_nzes2011 %>%   
 select(jpartyvote, jdiffvoting, \_singlefav)

We know that select() is a valid dplyr function, so that cannot be the problem. This means the problem might be the variable names. The issue is that R has rules about what variable names are legal (e.g. no spaces, starting with a letter) and when data is loaded, R will often fix variable names to make them legal. This happened to the \_singlefav at the time of loading the data.

We could check this by looking through every single variable name in the data with the names() command.

names(selected\_nzes2011)

## [1] "Jelect" "jblogel" "jnewspaper" "jnatradio"   
## [5] "jtalkback" "jdiscussp" "jrallies" "jpersuade"   
## [9] "jpcmoney" "jpcposter" "jlablike" "jnatlike"   
## [13] "jgrnlike" "jnzflike" "jactlike" "junflike"   
## [17] "jmaolike" "jmnplike" "jmostlike" "jmostlikex"   
## [21] "jrepublic" "jsphealth" "jspedu" "jspunemp"   
## [25] "jspdefence" "jspsuper" "jspbusind" "jsppolice"   
## [29] "jspwelfare" "jspenviro" "jgovpdk" "jgovplab"   
## [33] "jgovpnat" "jgovpgrn" "jgovpnzf" "jgovpact"   
## [37] "jgovunf" "jgovpmao" "jgovpmnp" "jnevervoteno"   
## [41] "jnevervotelab" "jnevervotenat" "jnevervotegrn" "jnevervotenzf"   
## [45] "jnevervoteact" "jnevervoteunf" "jnevervotemao" "jnevervotemnp"   
## [49] "jnevervoteoth" "jnevervoteothx" "jfirstpx" "jsecondp"   
## [53] "jage" "jlanguage" "jlanguagex" "jrollsex"   
## [57] "jhqual" "jwkft" "jwkpt" "jwkun"   
## [61] "jwkret" "jwkdis" "jwksch" "jwkunpo"   
## [65] "jwkunpi" "jhhincome" "jhhadults" "jhhchn"   
## [69] "jmarital" "r\_jind" "jlablr" "jnatlr"   
## [73] "jgrnlr" "jnzflr" "jactlr" "junflr"   
## [77] "jmaolr" "jmnplr" "jslflr" "jrelservices"   
## [81] "jrelnone" "jrelang" "jrelpres" "jrelcath"   
## [85] "jrelmeth" "jrelbap" "jrellat" "jrelrat"   
## [89] "jrelfun" "jrelothc" "jrelnonc" "jreligionx"   
## [93] "jreligiousity" "jethnicity\_e" "jethnicity\_m" "jethnicity\_p"   
## [97] "jethnicity\_a" "jethnicity\_o" "jethnicityx" "jethnicmost"   
## [101] "jethnicmostx" "jpartyvote" "jelecvote" "njptyvote"   
## [105] "njelecvote" "jdiffvoting" "X\_singlefav"

However, when we have hundreds of column names, a useful tip is to just search out only possible names. We can search the names for a fragment of the name by using the grep("FRAGMENT", variable, value = TRUE) command, which in this case might be:

grep("singlefav", names(selected\_nzes2011), value = TRUE)

## [1] "X\_singlefav"

The value = TRUE argument, as described in the help for the grep() function reports the mathing character string, as opposed to the index number for that string.

We can now confirm that the variable is called X\_singlefav, so that is how we should be referring to it.

selected\_nzes2011 %>%   
 select(jpartyvote, jdiffvoting, X\_singlefav) %>%   
 str()

These are all categorical data, however they are recorded as characters (text strings) as opposed to factors.

An easy way of tabulating these data to see how many times each level of is to use the group\_by() function along with the summarise() command:

selected\_nzes2011 %>%   
 group\_by(jpartyvote) %>%   
 summarise(count = n())

## # A tibble: 14 x 2  
## jpartyvote count  
## <chr> <int>  
## 1 Act 29  
## 2 ALC 10  
## 3 Alliance 2  
## 4 Another party 8  
## 5 Conservative 74  
## 6 Don't know 23  
## 7 Green 348  
## 8 Labour 749  
## 9 Mana 62  
## 10 Maori Party 128  
## 11 National 1130  
## 12 NZ First 216  
## 13 United Future 14  
## 14 <NA> 308

We can see that 23 people answered "Don't know". Since our question is about people who knew which party they voted for, we might want to exclude these observations from our analysis. We can do so by filtering them out.

selected\_nzes2011 %>%   
 filter(jpartyvote != "Don't know") %>%  
 group\_by(jpartyvote) %>%   
 summarise(count = n())

## # A tibble: 12 x 2  
## jpartyvote count  
## <chr> <int>  
## 1 Act 29  
## 2 ALC 10  
## 3 Alliance 2  
## 4 Another party 8  
## 5 Conservative 74  
## 6 Green 348  
## 7 Labour 749  
## 8 Mana 62  
## 9 Maori Party 128  
## 10 National 1130  
## 11 NZ First 216  
## 12 United Future 14

Because there is a %>% at the end of the line, R knows to continue on to the next line, as with any other 'to be continued' symbol at the end of the line.

Note that adding the filter also got rid of the NA entries. NA (Not Available) is used to indicate blank entries -- those observations for which there is no data recorded. It is always a good plan to be aware of NAs and deliberately include them in or exclude them from the analysis so that the final results are not surprising. In this case since NA indicates that these people did not answer the question about which party they voted for, exluding them from the analysis makes sense.

We can also similarly view the levels and number of occurances of these levels in the X\_singlefav variable:

selected\_nzes2011 %>%   
 group\_by(X\_singlefav) %>%   
 summarise(count = n())

## # A tibble: 8 x 2  
## X\_singlefav count  
## <chr> <int>  
## 1 Act 33  
## 2 Green 388  
## 3 Labour 1043  
## 4 Mana 47  
## 5 National 1266  
## 6 NZ First 138  
## 7 United Future 128  
## 8 <NA> 58

This set also has NA entries, but in this case we don't want to get rid of anything but the NAs so we need to target them directly. NA entries need special targeting because they do not actually exist (they are different to the text "NA" or a variable saved with the name NA).

If we only wanted to find the NAs we would use the is.na() function with the name of the variable inside the parentheses.

However since we want the entries that are **not** NAs we can use the **Not** operator, !, to indicate "we want all the ones that are not NA":!is.na(). Hence we can filter out all non NAs in our dplyr chain:

selected\_nzes2011 %>%   
 filter(!is.na(X\_singlefav)) %>%  
 group\_by(X\_singlefav) %>%   
 summarise(count = n())

## # A tibble: 7 x 2  
## X\_singlefav count  
## <chr> <int>  
## 1 Act 33  
## 2 Green 388  
## 3 Labour 1043  
## 4 Mana 47  
## 5 National 1266  
## 6 NZ First 138  
## 7 United Future 128

And remember that we can filter for multiple characteristics at once:

selected\_nzes2011 %>%   
 filter(!is.na(X\_singlefav), jpartyvote != "Don't know") %>%  
 group\_by(X\_singlefav) %>%   
 summarise(count=n())

## # A tibble: 7 x 2  
## X\_singlefav count  
## <chr> <int>  
## 1 Act 29  
## 2 Green 354  
## 3 Labour 914  
## 4 Mana 42  
## 5 National 1172  
## 6 NZ First 119  
## 7 United Future 115

If we examine the categories in jdiffvoting we can see that this variable has levels such as both "Don't know" and NA.

selected\_nzes2011 %>%   
 group\_by(jdiffvoting) %>%   
 summarise(count = n())

## # A tibble: 7 x 2  
## jdiffvoting count  
## <chr> <int>  
## 1 Don't know 63  
## 2 Voting can make a big difference to what happens 1605  
## 3 Voting can make a reasonable amount of difference to what happens 841  
## 4 Voting can make some difference to what happens 339  
## 5 Voting won't make any difference to what happens 119  
## 6 Voting won't make much difference to what happens 106  
## 7 <NA> 28

We need to decide how we want to handle these levels in our analysis.

Remember that our main question is about whether people vote for their favorite party or a diffent one. Hence an straighforwrd approach would be to first determine whether each observation in the data represents a person who voted for the party same as their favorite party or different. This requires creating a new variable with the mutate() function.

In creating this variable we want to evaluate if for a given observation the values in the jpartyvote and X\_singlefav variables are the same, or different:

selected\_nzes2011 <- selected\_nzes2011 %>%  
 mutate(sameparty = ifelse(jpartyvote == X\_singlefav, "same", "different"))

This creates a new variable named sameparty that has the value "same" if jpartyvote is equal to X\_singlefav, and "different" otherwise.

We can again check our work by exploring the groupings in a View:

selected\_nzes2011 %>%   
group\_by(jpartyvote, X\_singlefav, sameparty) %>%  
 summarise(count = n())

## `summarise()` has grouped output by 'jpartyvote', 'X\_singlefav'. You can override using the `.groups` argument.

## # A tibble: 82 x 4  
## # Groups: jpartyvote, X\_singlefav [82]  
## jpartyvote X\_singlefav sameparty count  
## <chr> <chr> <chr> <int>  
## 1 Act Act same 12  
## 2 Act Green different 1  
## 3 Act National different 14  
## 4 Act United Future different 1  
## 5 Act <NA> <NA> 1  
## 6 ALC Green different 1  
## 7 ALC Labour different 4  
## 8 ALC National different 2  
## 9 ALC United Future different 3  
## 10 Alliance Labour different 1  
## # ... with 72 more rows

We can see that observations where jpartyvote equaled X\_singlefav, the value "same" was recorded for the new variable sameparty, and the value "different" was recorded otherwise. If either jpartyvote or X\_singlefav had an NA, R could not check for equality and hence NA was recorded for the sameparty variable as well.

To view and summarize the "same" entries we can use the following:

selected\_nzes2011 %>%   
 group\_by(jpartyvote, X\_singlefav, sameparty) %>%  
 summarise(count = n()) %>%   
 filter(sameparty == "same")

## `summarise()` has grouped output by 'jpartyvote', 'X\_singlefav'. You can override using the `.groups` argument.

## # A tibble: 7 x 4  
## # Groups: jpartyvote, X\_singlefav [7]  
## jpartyvote X\_singlefav sameparty count  
## <chr> <chr> <chr> <int>  
## 1 Act Act same 12  
## 2 Green Green same 237  
## 3 Labour Labour same 632  
## 4 Mana Mana same 31  
## 5 National National same 1004  
## 6 NZ First NZ First same 82  
## 7 United Future United Future same 5

And to view and summarize the "different" entries we can use the following:

selected\_nzes2011 %>%   
 group\_by(jpartyvote, X\_singlefav, sameparty) %>%  
 summarise(count = n()) %>%   
 filter(sameparty == "different")

## `summarise()` has grouped output by 'jpartyvote', 'X\_singlefav'. You can override using the `.groups` argument.

## # A tibble: 59 x 4  
## # Groups: jpartyvote, X\_singlefav [59]  
## jpartyvote X\_singlefav sameparty count  
## <chr> <chr> <chr> <int>  
## 1 Act Green different 1  
## 2 Act National different 14  
## 3 Act United Future different 1  
## 4 ALC Green different 1  
## 5 ALC Labour different 4  
## 6 ALC National different 2  
## 7 ALC United Future different 3  
## 8 Alliance Labour different 1  
## 9 Alliance National different 1  
## 10 Another party Green different 2  
## # ... with 49 more rows

We can also check how we got any NAs we have by using the is.na() function:

selected\_nzes2011 %>%   
 group\_by(jpartyvote, X\_singlefav, sameparty) %>%  
 summarise(count = n()) %>%   
 filter(is.na(sameparty))

## `summarise()` has grouped output by 'jpartyvote', 'X\_singlefav'. You can override using the `.groups` argument.

## # A tibble: 16 x 4  
## # Groups: jpartyvote, X\_singlefav [16]  
## jpartyvote X\_singlefav sameparty count  
## <chr> <chr> <chr> <int>  
## 1 Act <NA> <NA> 1  
## 2 Conservative <NA> <NA> 1  
## 3 Don't know <NA> <NA> 7  
## 4 Green <NA> <NA> 1  
## 5 Labour <NA> <NA> 11  
## 6 Maori Party <NA> <NA> 2  
## 7 National <NA> <NA> 7  
## 8 NZ First <NA> <NA> 2  
## 9 <NA> Act <NA> 4  
## 10 <NA> Green <NA> 32  
## 11 <NA> Labour <NA> 121  
## 12 <NA> Mana <NA> 4  
## 13 <NA> National <NA> 92  
## 14 <NA> NZ First <NA> 17  
## 15 <NA> United Future <NA> 12  
## 16 <NA> <NA> <NA> 26

The checks show that the observations with NAs in the samepartyare going to be excluded from the analysis when we fiter out the NAs in the jpartyvote and X\_singlefav variables, so we don't need to worry about them anymore.

## Step four. Prepare for the second question

As a second question, we might be interested in exploring the relationship between age of voters and how much they like the NZ First party. We become familiar with the variables jnzflike and jage in the codebook, then explore the data.

str(selected\_nzes2011$jnzflike)

## Factor w/ 12 levels "0","1","10","2",..: 1 1 4 10 4 11 NA NA 1 12 ...

str(selected\_nzes2011$jage)

## int [1:3101] 37 37 28 71 43 NA 59 68 64 70 ...

jnzflike is a factor variable, in fact it's ordinal and by default the levels are listed in alphabetical order. Since this is a categorical variable, we can also summarize the occurances of each level with group\_by() and summarise() again:

selected\_nzes2011 %>%   
 group\_by(jnzflike) %>%   
 summarise(count = n())

## # A tibble: 13 x 2  
## jnzflike count  
## <fct> <int>  
## 1 0 622  
## 2 1 298  
## 3 10 134  
## 4 2 266  
## 5 3 227  
## 6 4 162  
## 7 5 544  
## 8 6 165  
## 9 7 138  
## 10 8 107  
## 11 9 81  
## 12 Don't know 224  
## 13 <NA> 133

While jnzflike is on a 0 to 10 scale, this variable also has a level labeled "Don't know", which is why R stores this variable as not a numeric variable.

jage, on the other hand, is an integer, with values that are whole numbers between 0 and infinity (or NA). For this variable we would want to take a look at numerical summaries such as means, medians, etc.

selected\_nzes2011 %>%   
 summarise(agemean = mean(jage), agemedian = median(jage), agesd = sd(jage),   
 agemin = min(jage), agemax = max(jage))

## agemean agemedian agesd agemin agemax  
## 1 NA NA NA NA NA

What went wrong? The reason why all of the results were reported as NAs is that there were some NA entries in the jage variable (people not reporting their age). Since it is not possible to take the average of a series of values that contain NAs, obtaining the numerical summaries requires that we exclude the NAs from the calculation.

Most numerical summary functions allow us to easily exclude NAs with the na.rm argument. See the help documentation for the median function for more information.

?median

## starting httpd help server ... done

An alternative approach is just to filter out the NAs first, and then ask for the numerical summaries:

selected\_nzes2011 %>%   
 filter(!(is.na(jage))) %>%  
 summarise(agemean = mean(jage), agemedian = median(jage), agesd = sd(jage),   
 agemin = min(jage), agemax = max(jage))

## agemean agemedian agesd agemin agemax  
## 1 53.22328 54 17.5371 18 100

An age range of 18 to 100 is a reasonable age range for a voting age population, so there are no obvious errors in the data. If there were, we would need to decide if we should filter them out of the analysis.

Having gained some familiarity with the specific variables we are using, we next need to consider if there is additional work we should do on the data in investigating the question. There are a number of different approaches we might take. For example, we could consider if those that strongly like NZ First are older than those that strongly dislike NZ First, or we could consider if old people like NZ First more than young people.

### Approach 1: Strongly liking and disliking NZ First and age

If we wanted to select only two of the possible levels in how much people like NZ First, we can filter for these specific levels. When interested in filtering for multiple values a variable can take, the %in% operator can come in handy:

selected\_nzes2011 %>%   
 filter(jnzflike %in% c("0","10")) %>%  
 group\_by(jnzflike) %>%   
 summarise(count = n())

## # A tibble: 2 x 2  
## jnzflike count  
## <fct> <int>  
## 1 0 622  
## 2 10 134

Remember that the jnzflike is not a numerical variable, hence we use the quotation marks around the values (even though they happen to be numbers).

This is an example of simpligying the analysis by considering only two levels of a categorical variable, as opposed to all possible levels.

### Approach 2: Age and liking for NZ First

We might also like to refine our question slightly, asking do people above retirement age (65 in New Zealand) like NZ First more than younger people. To do this we can turn the numeric age variable into a categorical variable based on whether people are 65 years or older or younger than 65. Once again we make use of the mutate() and ifelse() functions:

selected\_nzes2011 <- selected\_nzes2011 %>%   
 mutate(retiredage = ifelse(jage >= 65, "retired age", "working age"))  
selected\_nzes2011 %>%   
 group\_by(retiredage) %>%   
 summarise(count = n())

## # A tibble: 3 x 2  
## retiredage count  
## <chr> <int>  
## 1 retired age 876  
## 2 working age 2156  
## 3 <NA> 69

We can see that individuals in the dataset are now labeled as either "retired age" or "working age" or neither (NA), which we can easily filter out if need be.

This is an example of using a numerical threshold to convert a numerical variable to a categorical variable.

For approach 2, we might also be want to turn the scale of liking into numeric values, because at the moment we cannot easily get summary information of the data in factor form. For example, if we ty to run the following command, we get an error saying "need numeric data".

selected\_nzes2011 %>%   
 group\_by(retiredage) %>%   
 summarise(medlike = median(jnzflike))

it generates a "need numeric data" error.

We can change the type of data with functions of the form as.thingtochangeto(), but it is easy to go wrong with factors. For example, this is wrong:

selected\_nzes2011 <- selected\_nzes2011 %>%   
 mutate(numlikenzf = as.numeric(jnzflike))

We can see it has gone wrong if we use grouping to check our work (and it is a very good plan to check our work after converting factors).

selected\_nzes2011 %>%   
 group\_by(jnzflike, numlikenzf) %>%   
 summarise(count = n())

## `summarise()` has grouped output by 'jnzflike'. You can override using the `.groups` argument.

## # A tibble: 13 x 3  
## # Groups: jnzflike [13]  
## jnzflike numlikenzf count  
## <fct> <dbl> <int>  
## 1 0 1 622  
## 2 1 2 298  
## 3 10 3 134  
## 4 2 4 266  
## 5 3 5 227  
## 6 4 6 162  
## 7 5 7 544  
## 8 6 8 165  
## 9 7 9 138  
## 10 8 10 107  
## 11 9 11 81  
## 12 Don't know 12 224  
## 13 <NA> NA 133

Factor entries have two parts: the text we see on the screen, and a numeric order (remember how 10 was coming between 1 and 2 because of the alphabetical order). When we say "turn this into a number", R uses the numeric order in which it stores the values to do that conversion, as opposed to the names of the levels of the categorical variable. Hence, we need a conversion method that will use the text strings that label the levels, as opposed to the storage order of these levels. We can do this by first saving the variable as a character variable, and then turning it into a number:

selected\_nzes2011 <- selected\_nzes2011 %>%   
 mutate(numlikenzf = as.numeric(as.character(jnzflike)))

## Warning in mask$eval\_all\_mutate(quo): NAs introduced by coercion

The warning "NAs introduced by coercion" happens since the level "Don't know" cannot be turned into a number. But this should be fine for our purposes since we are interested in the numerical responses anyway.

selected\_nzes2011 %>%   
 group\_by(jnzflike, numlikenzf) %>%   
 summarise(count = n())

## `summarise()` has grouped output by 'jnzflike'. You can override using the `.groups` argument.

## # A tibble: 13 x 3  
## # Groups: jnzflike [13]  
## jnzflike numlikenzf count  
## <fct> <dbl> <int>  
## 1 0 0 622  
## 2 1 1 298  
## 3 10 10 134  
## 4 2 2 266  
## 5 3 3 227  
## 6 4 4 162  
## 7 5 5 544  
## 8 6 6 165  
## 9 7 7 138  
## 10 8 8 107  
## 11 9 9 81  
## 12 Don't know NA 224  
## 13 <NA> NA 133

Converting the factor to a character first ensures that the numerical values used in the labels of the levels of the categorical variable are used.

Now that we cleaned up the data in a way that addresses the needs of the research questions we want to explore, we are ready to continue with our analysis.

## Appendix: List of fields in example data

|  |  |  |
| --- | --- | --- |
| Variable | Question | DataType |
| jactlike | A14: how much like Act | Factor |
| jactlr | A18: Act on left-right scale | chr |
| jage | Respondent's age in years | int |
| jblogel | A6h: visit political blog for election | chr |
| jdiffvoting | A13: does voting make any difference to what happens | chr |
| jdiscussp | A11a: how often discussed politics with others | chr |
| Jelect | Electorate | int |
| jelecvote | C4: if cast electorate vote, for which party's candidate | chr |
| jethnicity\_a | F19a: ethnicity - Asian | chr |
| jethnicity\_e | F19a: ethnicity - NZ European | chr |
| jethnicity\_m | F19a: ethnicity - NZ Maori | chr |
| jethnicity\_o | F19a: ethnicity - Other | chr |
| jethnicity\_p | F19a: ethnicity - Pacific | chr |
| jethnicityx | F19ax: other ethnic group belonged to detail | chr |
| jethnicmost | F19b: ethnic group identified with most | chr |
| jethnicmostx | F19bx: other ethnic group identified with most | chr |
| jfirstpx | C10x: on election day other party most wanted to be in government | chr |
| jgovpact | C8: Act helped form the government after 2008 election | chr |
| jgovpdk | C8: can't recall which parties formed the government after 2008 election | chr |
| jgovpgrn | C8: Greens helped form the government after 2008 election | chr |
| jgovplab | C8: Labour helped form the government after 2008 election | chr |
| jgovpmao | C8: Maori Party helped form the government after 2008 election | chr |
| jgovpmnp | C8: Mana Party helped form the government after 2008 election | chr |
| jgovpnat | C8: National helped form the government after 2008 election | chr |
| jgovpnzf | C8: NZ First helped form the government after 2008 election | chr |
| jgovunf | C8: United Future helped form the government after 2008 election | chr |
| jgrnlike | A14: how much like Greens | Factor |
| jgrnlr | A18: Greens on left-right scale | chr |
| jhhadults | F23a: number of adults in household | int |
| jhhchn | F23b: number of children in household | int |
| jhhincome | F22: household income between 1 April 2010 and 31 March 2011 | chr |
| jhqual | F8: highest formal educational qualification | chr |
| jlablike | A14: how much like Labour | Factor |
| jlablr | A18: Labour on left-right scale | chr |
| jlanguage | F3: main language spoken at your home | chr |
| jlanguagex | F3x: other main language spoken | chr |
| jmaolike | A14: how much like Maori Party | Factor |
| jmaolr | A18: Maori Party on left-right scale | chr |
| jmarital | F24: marital status | chr |
| jmnplike | A14: how much like Mana Party | Factor |
| jmnplr | A18: Mana Party on left-right scale | chr |
| jmostlike | A15: on election day which party liked most | chr |
| jmostlikex | A15x: other party liked most | chr |
| jnatlike | A14: how much like National | Factor |
| jnatlr | A18: National on left-right scale | chr |
| jnatradio | A10d: how often followed election news on Radio New Zealand: National | chr |
| jnevervoteact | C16: would never vote for Act | chr |
| jnevervotegrn | C16: would never vote for Greens | chr |
| jnevervotelab | C16: would never vote for Labour | chr |
| jnevervotemao | C16: would never vote for Maori Party | chr |
| jnevervotemnp | C16: would never vote for Mana Party | chr |
| jnevervotenat | C16: would never vote for National | chr |
| jnevervoteno | C16: no party for which you would never vote | chr |
| jnevervotenzf | C16: would never vote for NZ First | chr |
| jnevervoteoth | C16: would never vote for another party | chr |
| jnevervoteothx | C16: other party for for which you would never vote | chr |
| jnevervoteunf | C16: would never vote for United Future | chr |
| jnewspaper | A10c: how often followed election news in newspaper | chr |
| jnzflike | A14: how much like NZ First | Factor |
| jnzflr | A18: NZ First on left-right scale | chr |
| jpartyvote | C3: if cast party vote, for which party | chr |
| jpcmoney | A11d: how often contributed money to a party or candidate | chr |
| jpcposter | A11e: how often put up party or candidate posters | chr |
| jpersuade | A11c: how often talk to anyone to persuade them how to vote | chr |
| jrallies | A11b: how often attended political meetings or rallies | chr |
| jrelang | F17: anglican | chr |
| jrelbap | F17: baptist | chr |
| jrelcath | F17: catholic | chr |
| jrelfun | F17: independent-fundamentalist-pentecostal church | chr |
| jreligionx | F17x: other religion detail | chr |
| jreligiousity | F18: how religious are you | chr |
| jrellat | F17: latter day saints | chr |
| jrelmeth | F17: methodist | chr |
| jrelnonc | F17: non-Christian | chr |
| jrelnone | F17: no religion | chr |
| jrelothc | F17: other Christian | chr |
| jrelpres | F17: presbyterian | chr |
| jrelrat | F17: ratana | chr |
| jrelservices | F16: apart from weddings, funerals, baptisms, how often do you attend religious services | chr |
| jrepublic | B1: should NZ become a republic or retain Queen as head of state | chr |
| jrollsex | Respondent's gender from electoral roll | chr |
| jsecondp | C11: on election day which party overall was you second choice to be in government | chr |
| jslflr | A19: yourself on left-right scale | chr |
| jspbusind | B3f: should there be more or less public spending on business and industry | chr |
| jspdefence | B3d: should there be more or less public spending on defence | chr |
| jspedu | B3b: should there be more or less public spending on education | chr |
| jspenviro | B3i: should there be more or less public spending on the environment | chr |
| jsphealth | B3a: should there be more or less public spending on health | chr |
| jsppolice | B3g: should there be more or less public spending on police and law enforcement | chr |
| jspsuper | B3e: should there be more or less public spending on superannuation | chr |
| jspunemp | B3c: should there be more or less public spending on unemployment benefits | chr |
| jspwelfare | B3h: should there be more or less public spending on welfare benefits | chr |
| jtalkback | A10e: how often followed election news on talkback radio | chr |
| junflike | A14: how much like United Future | Factor |
| junflr | A18: United Future on left-right scale | chr |
| jwkdis | F9: disabled, unable to work | chr |
| jwkft | F9: working full-time for pay or other income | chr |
| jwkpt | F9: working part-time for pay or other income | chr |
| jwkret | F9: retired | chr |
| jwksch | F9: at school, university, or other educational institution | chr |
| jwkun | F9: unemployed, laid off, looking for work | chr |
| jwkunpi | F9: working unpaid within the home | chr |
| jwkunpo | F9: working unpaid outside the home | chr |
| njelecvote | Electorate Vote with nonvote | chr |
| njptyvote | Party Vote with nonvote | chr |
| r\_jind | Respondent Industry Codes | chr |
| \_singlefav | Caluclated Variable of most liked of major parties Question A14 | chr |