Descriptive Statistics in R

Jan Rovny

Basic Data Descriptions

To find out about a dataset in R we can run a names command on the particular dataframe (dataset). Let's return to our example dataset, first loading it into R:

```
library(rio)
D<-import("https://jan-rovny.squarespace.com/s/France.dta")
```

Let's explore our data:

```
names(D)
```

```
 [1] "dpt"             "female"          "age"             "educ"
 [5] "inc"             "urban"           "religion"        "religiosity"
 [9] "lr"              "redist"          "feminism"        "pro_immig"
[13] "adopt_homo"      "anti_vax"        "gilets"          "elit_oppose_rur"
[17] "sov_eu"          "str_lead"        "milit"           "prvt_LFI"
[21] "prvt_PS"         "prvt_EELV"       "prvt_LREM"       "prvt_LR"
[25] "prvt_FN"         "prvt_RE"         "vote1"
```

This produces a list of all the variable names in the dataset. Alternatively, we can ask to receive both variable names and some basic summary statistics:

```
summary(D)
```

```
      dpt            female            age             educ
 Min.   : 1.00   Min.   :0.0000   Min.   :18.00   Min.   :0.00
 1st Qu.:33.00   1st Qu.:0.0000   1st Qu.:27.00   1st Qu.:4.00
 Median :59.00   Median :1.0000   Median :39.00   Median :6.00
 Mean   :53.93   Mean   :0.5331   Mean   :44.03   Mean   :5.39
```

```
3rd Qu.:75.00   3rd Qu.:1.0000   3rd Qu.:59.00   3rd Qu.:7.00
Max.   :95.00   Max.   :2.0000   Max.   :93.00   Max.   :8.00
NA's   :5
     inc             urban          religion      religiosity
Min.   : 1.000   Min.   :1.000   Min.   :1.000   Min.   :0.000
1st Qu.: 6.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
Median : 8.000   Median :2.000   Median :4.000   Median :1.000
Mean   : 7.939   Mean   :1.888   Mean   :4.171   Mean   :1.554
3rd Qu.:10.000   3rd Qu.:2.000   3rd Qu.:7.000   3rd Qu.:2.000
Max.   :13.000   Max.   :4.000   Max.   :7.000   Max.   :4.000
NA's   :140      NA's   :10      NA's   :46      NA's   :836
      lr            redist         feminism      pro_immig      adopt_homo
Min.   :1.000   Min.   :1.000   Min.   :1.00   Min.   :1.00   Min.   :0.000
1st Qu.:2.000   1st Qu.:1.000   1st Qu.:2.00   1st Qu.:2.00   1st Qu.:2.000
Median :3.000   Median :2.000   Median :3.00   Median :3.00   Median :2.000
Mean   :2.912   Mean   :2.297   Mean   :2.68   Mean   :2.72   Mean   :2.093
3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:4.00   3rd Qu.:4.00   3rd Qu.:3.000
Max.   :5.000   Max.   :4.000   Max.   :4.00   Max.   :4.00   Max.   :3.000
NA's   :78      NA's   :58      NA's   :73     NA's   :91     NA's   :56
   anti_vax          gilets       elit_oppose_rur     sov_eu
Min.   :0.0000   Min.   :0.0000   Min.   :0.000   Min.   :1.000
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:2.000
Median :0.0000   Median :0.0000   Median :1.000   Median :3.000
Mean   :0.5662   Mean   :0.7594   Mean   :1.356   Mean   :2.535
3rd Qu.:2.0000   3rd Qu.:2.0000   3rd Qu.:2.000   3rd Qu.:3.000
Max.   :2.0000   Max.   :2.0000   Max.   :3.000   Max.   :4.000
NA's   :46       NA's   :51       NA's   :220     NA's   :145
   str_lead          milit           prvt_LFI         prvt_PS
Min.   :0.0000   Min.   :0.0000   Min.   : 1.000   Min.   : 1.000
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 1.000   1st Qu.: 1.000
Median :0.0000   Median :0.0000   Median : 5.000   Median : 5.000
Mean   :0.5971   Mean   :0.4753   Mean   : 5.135   Mean   : 4.809
3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 8.000   3rd Qu.: 7.000
Max.   :3.0000   Max.   :3.0000   Max.   :11.000   Max.   :11.000
NA's   :64       NA's   :933      NA's   :83       NA's   :97
  prvt_EELV        prvt_LREM         prvt_LR          prvt_FN
Min.   : 1.000   Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
1st Qu.: 3.000   1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 1.000
Median : 6.000   Median : 6.000   Median : 5.000   Median : 1.000
Mean   : 5.565   Mean   : 5.284   Mean   : 4.576   Mean   : 3.924
3rd Qu.: 8.000   3rd Qu.: 8.000   3rd Qu.: 6.000   3rd Qu.: 7.000
Max.   :11.000   Max.   :11.000   Max.   :11.000   Max.   :11.000
NA's   :68       NA's   :98       NA's   :113      NA's   :81
```

```
   prvt_RE           vote1
Min.   : 1.000   Min.   :1.000
1st Qu.: 1.000   1st Qu.:1.000
Median : 1.000   Median :3.000
Mean   : 2.406   Mean   :2.939
3rd Qu.: 3.000   3rd Qu.:5.000
Max.   :11.000   Max.   :6.000
NA's   :63       NA's   :575
```

We can also ask about the nature of each variable by typing:

```
is.character(D$vote1)
```

```
[1] FALSE
```

```
is.numeric(D$age)
```

```
[1] TRUE
```

```
is.factor(D$religion)
```

```
[1] FALSE
```

```
is.integer(D$female)
```

```
[1] FALSE
```

```
is.vector(D$lr)
```

```
[1] FALSE
```

R answers TRUE or FALSE.

Descriptive Statistics

Next, we should run some descriptive statistics on our data. Descriptive statistics do not test any hypotheses and do not try to infer any general rules from the data. They simply describe the data we have in front of us. The most basic descriptive statistics are measures of central tendency, such as mean, mode and median, and measures of dispersion, such as variance and standard deviation.

Measure	Calculation	Description
Mean	$\bar{X} = \mu = \frac{\sum x_i}{N}$	the arithmetic mean
Mode		the most frequently occurring value
Median		the central value
Variance	$\sigma^2 = \frac{\sum (x_i - \bar{X})^2}{N-1}$	square deviation from mean
Standard Deviation	$\sigma = \sqrt{\frac{\sum (x_i - \bar{X})^2}{N-1}}$	deviation from mean

In R, we can easily obtain these measures:

```
summary(D$age)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  18.00   27.00   39.00   44.03   59.00   93.00
```

This gives us the minimum, maximum, mean and median values of age. If we want particular statistics, we can (at any point, even within other commands) ask R to produce them by issuing the following commands:

```
mean(D$age, na.rm=T)
```

```
[1] 44.02613
```

```
median(D$age, na.rm=T)
```

```
[1] 39
```

```
var(D$age, na.rm=T)
```

```
[1] 376.66
```

```
sd(D$age, na.rm=T)
```

```
[1] 19.40773
```

```
min(D$age, na.rm=T)
```

```
[1] 18
```

```
max(D$age, na.rm=T)
```

```
[1] 93
```

```
range(D$age, na.rm=T)
```

```
[1] 18 93
```

The mode of a vector is a little harder to obtain. To get the mode of vector x, you can get it like this:

```
names(sort(-table(D$age)))[1]
```

```
[1] "35"
```

This creates a table of the frequencies of each value, multipling by -1 and sorting puts the largest frequency first, and 'names()[1]' extracts the name of the first element, which is the sample mode. You may need to then use 'as.numeric()' on the result if you want a number.

Statistics Summary

There is a useful package for summarizing data, called 'modelsummary'. We can look at all our continuous variables by specifying

```
library(modelsummary)
datasummary_skim(D)
```

	Unique (#)	Missing (%)	Mean	SD	Min	Median	Max
dpt	95	0	53.9	26.5	1.0	59.0	95.0
female	3	0	0.5	0.5	0.0	1.0	2.0
age	76	0	44.0	19.4	18.0	39.0	93.0
educ	9	0	5.4	1.9	0.0	6.0	8.0
inc	14	8	7.9	2.6	1.0	8.0	13.0
urban	5	1	1.9	0.9	1.0	2.0	4.0
religion	8	3	4.2	2.8	1.0	4.0	7.0
religiosity	6	49	1.6	1.1	0.0	1.0	4.0
lr	6	5	2.9	1.0	1.0	3.0	5.0
redist	5	3	2.3	1.0	1.0	2.0	4.0
feminism	5	4	2.7	1.1	1.0	3.0	4.0
pro_immig	5	5	2.7	1.1	1.0	3.0	4.0
adopt_homo	5	3	2.1	1.0	0.0	2.0	3.0
anti_vax	4	3	0.6	0.9	0.0	0.0	2.0
gilets	4	3	0.8	0.9	0.0	0.0	2.0
elit_oppose_rur	5	13	1.4	1.0	0.0	1.0	3.0
sov_eu	5	8	2.5	1.0	1.0	3.0	4.0
str_lead	5	4	0.6	0.8	0.0	0.0	3.0
milit	5	54	0.5	0.8	0.0	0.0	3.0
prvt_LFI	12	5	5.1	3.6	1.0	5.0	11.0
prvt_PS	12	6	4.8	2.9	1.0	5.0	11.0
prvt_EELV	12	4	5.6	3.0	1.0	6.0	11.0
prvt_LREM	12	6	5.3	3.3	1.0	6.0	11.0
prvt_LR	12	7	4.6	2.9	1.0	5.0	11.0
prvt_FN	12	5	3.9	3.6	1.0	1.0	11.0
prvt_RE	12	4	2.4	2.7	1.0	1.0	11.0
vote1	7	33	2.9	1.7	1.0	3.0	6.0

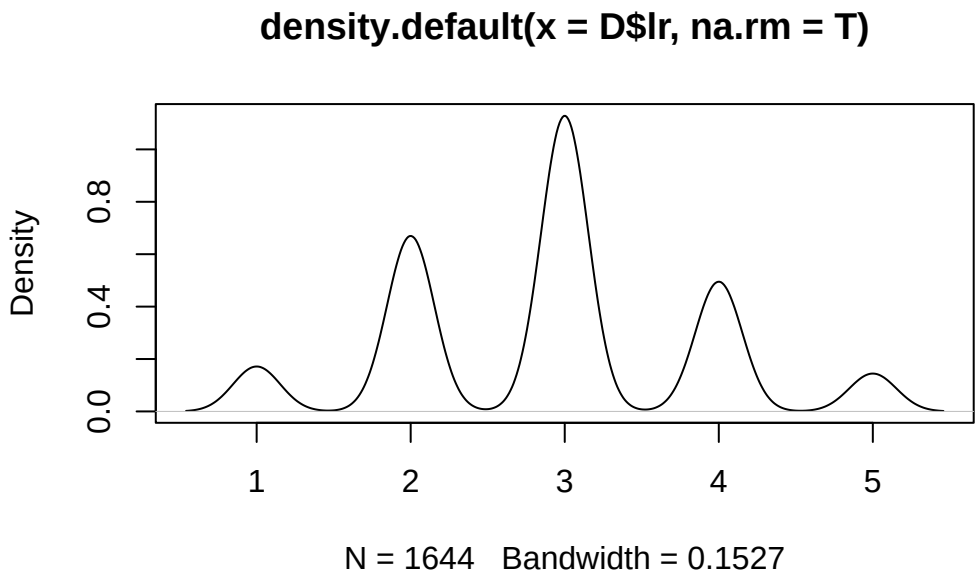
Basic plots

The subsequent step in learning about our data should be the plotting of the data. Mean and variance give us a good idea as to the central tendency and dispersion of a variable, but it is even more interesting to see the frequency distribution across its values. To see a distribution of of a variable we first need to create the distribution density function:

```
den<-density(D$lr, na.rm=T)
```

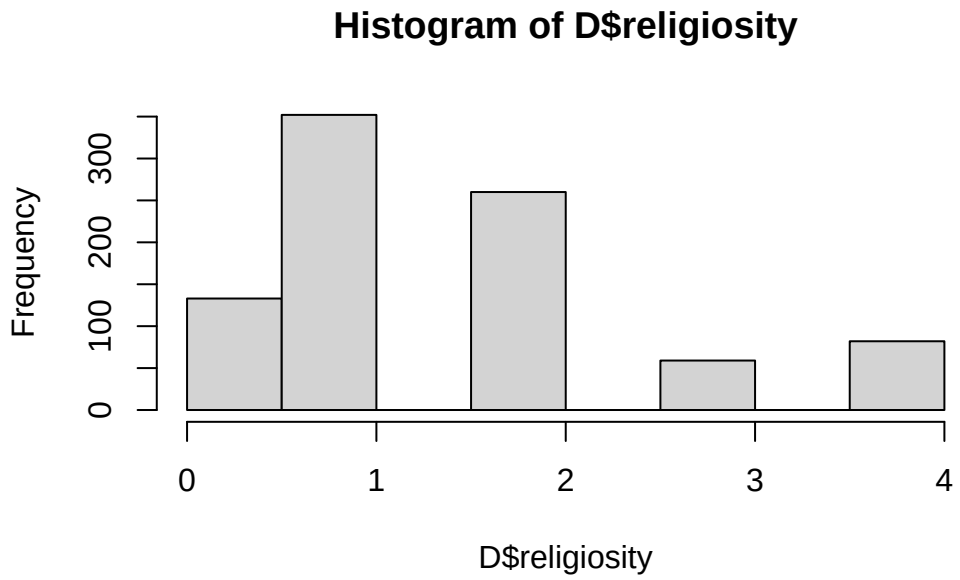
Now we can plot d to see the distribution:

```
plot(den)
```



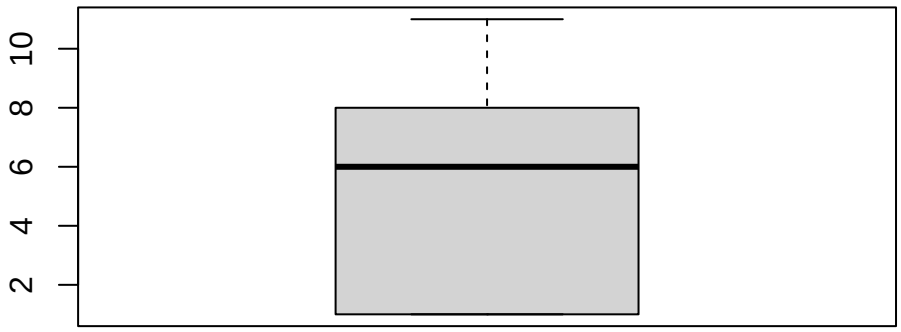
Density functions are, however, only meaningful for continuous data. In the cases of categorical or ordinal data it is more meaningful to look at a histogram. To plot a histogram in R, we simply say:

```
hist(D$religiosity)
```



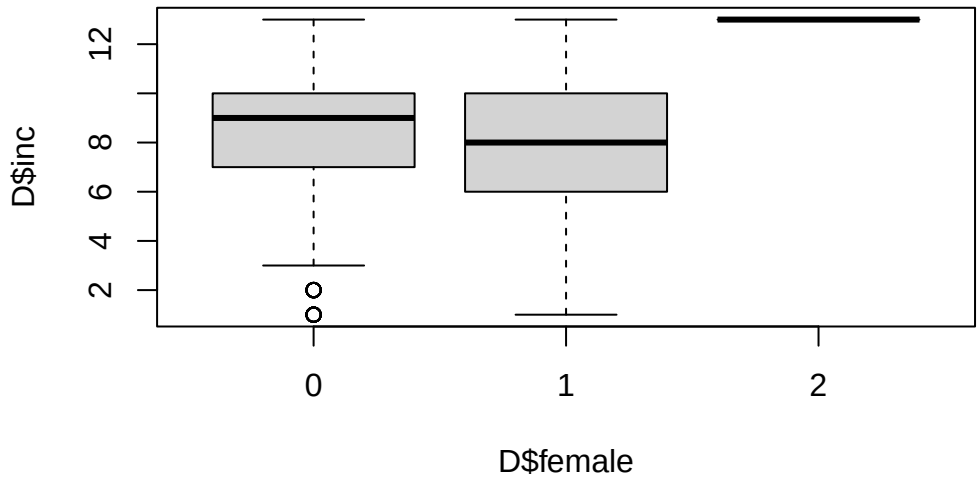
Another useful descriptive tool is a boxplot. A boxplot shows us the median, the quartiles, and the maximum and minimum of a variable. In R, say:

```
boxplot(D$prvt_LREM)
```



Boxplots are particularly useful for comparing the distributions of certain subsets of a given variable. Say that we are interested in seeing the different income distributions of men and women. We can do this by looking at a boxplot of income by gender:

```
boxplot(D$inc ~ D$female)
```



Nicer plots

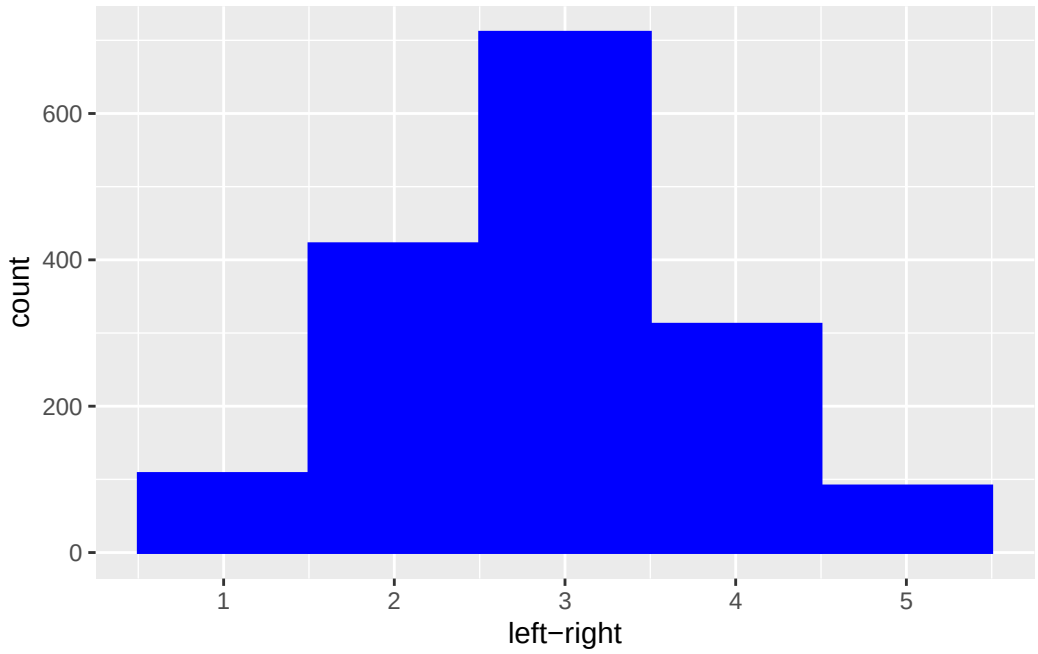
R is famous for being able to create pretty graphics. One of the most commonly used packages for this is 'ggplot2.' Let's see what we can do with it:

```
library(ggplot2) #call up the package
```

Now, let's create some nicer descriptive graphics. Let's look again at the distribution of left-right preferences:

```
ggplot(D,aes(lr)) +
  geom_histogram(binwidth = 1, color="blue", fill="blue")+ #determines type, arguments def
  xlab("left-right") #label x axis
```

```
Warning: Removed 78 rows containing non-finite values (`stat_bin()`).
```

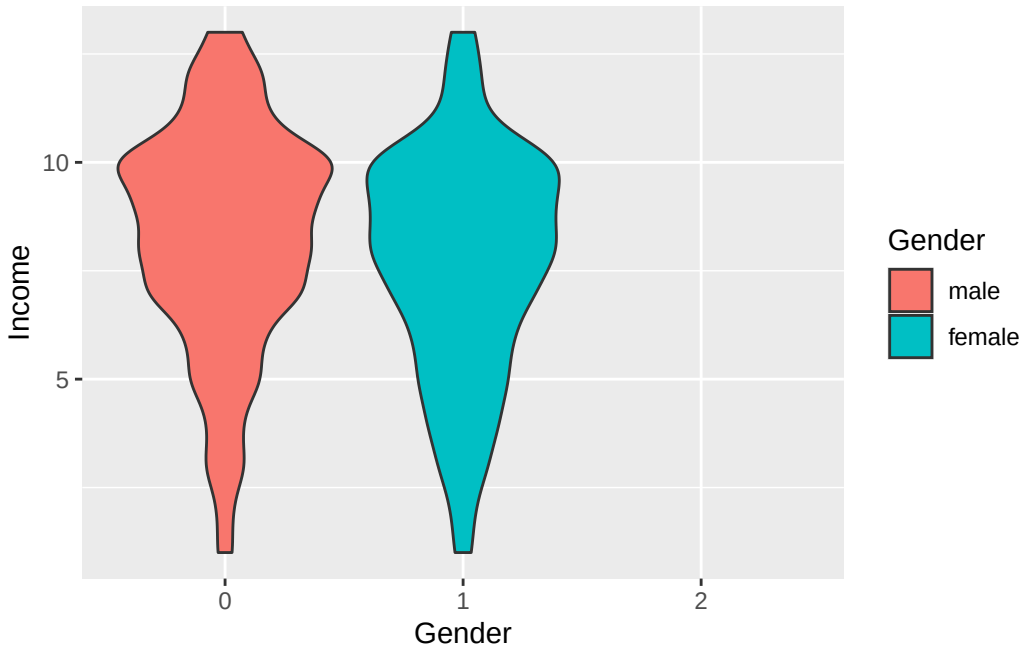


Let's consider again income and gender:

```
ggplot(D,aes(y=as.numeric(inc), x=as.factor(female), fill=as.factor(female)))+ #define axe
  geom_violin()+ #define violin plot
  xlab("Gender")+ylab("Income")+ #label axes
  scale_fill_discrete(name="Gender", labels=c("male","female")) #label fill title and labe
```

```
Warning: Removed 140 rows containing non-finite values (`stat_ydensity()`).
```

```
Warning: Groups with fewer than two data points have been dropped.
```



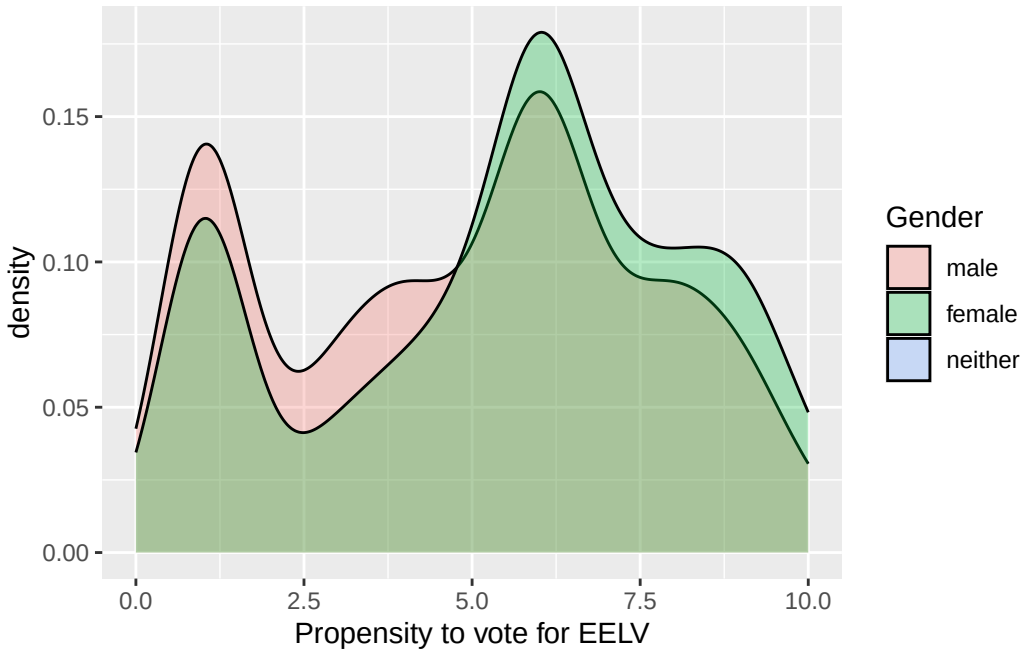
Next, let us look at the propensity of men and women to vote for the Green (EELV) party:

```
ggplot(D,aes(x=prvt_EELV, fill=as.factor(female)))+ #need to specify 'femal' as factor
  geom_density(alpha=0.3)+ #allows transparency, alpha defines level of opacity
  xlim(0,10)+ #defines range of x axis
  xlab("Propensity to vote for EELV")+ #x label
  scale_fill_discrete(name="Gender", labels=c("male","female", "neither")) #name and label
```

```
Warning: Removed 176 rows containing non-finite values (`stat_density()`).
```

```
Warning: Groups with fewer than two data points have been dropped.
```

```
Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
-Inf
```



Note that women are quite more likely to support the Greens.

Tables

Finally, it is very useful to organize our data into a table. A two-way table arranges the values of one variable by the values of another. Such organization is of course only meaningful for categorical or ordinal data, not for continuous variables. Let's make a table summarizing the vote for different candidates by gender (note, we are working with the variable *vote* created in the previous lesson dealing with Operations in R!):

```
table(D$female, D$vote)
```

```
  Melenchon Jadot Macron Peceresse Le Pen Zemmour
0       180    28    149        25    110      39
1       204    42    168        25    145      31
2         1     0      0         0      0       0
```

This, however, creates a table with raw numbers, which is not very useful. To do this comparison meaningfully, we must compare proportional data. In R we first create a raw table of vote by rich:

```
cand.table<-table(D$female,D$vote)
```

Now we use the prop.table command to create proportions:

```
prop.table(cand.table) # gives us the proportions by all cells
```

```
     Melenchon        Jadot       Macron    Peceresse       Le Pen
0 0.1569311247 0.0244115083 0.1299040976 0.0217959895 0.0959023540
1 0.1778552746 0.0366172624 0.1464690497 0.0217959895 0.1264167393
2 0.0008718396 0.0000000000 0.0000000000 0.0000000000 0.0000000000

       Zemmour
0 0.0340017437
1 0.0270270270
2 0.0000000000
```

```
prop.table(cand.table,1) #gives us the proportions by rows
```

```
   Melenchon      Jadot     Macron  Peceresse     Le Pen    Zemmour
0 0.33898305 0.05273070 0.28060264 0.04708098 0.20715631 0.07344633
1 0.33170732 0.06829268 0.27317073 0.04065041 0.23577236 0.05040650
2 1.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
```

```
prop.table(cand.table,2) #gives us the proportions by columns
```

```
    Melenchon       Jadot      Macron   Peceresse      Le Pen     Zemmour
0 0.467532468 0.400000000 0.470031546 0.500000000 0.431372549 0.557142857
1 0.529870130 0.600000000 0.529968454 0.500000000 0.568627451 0.442857143
2 0.002597403 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
```

In order to test whether there is a difference gender support across parties, we can ask R to provide the χ^2 test:

```
library(MASS) #load the appropriate library
chisq.test(cand.table) #perform the test
```

```
Warning in chisq.test(cand.table): Chi-squared approximation may be incorrect
```

```
    Pearson's Chi-squared test

data:  cand.table
X-squared = 7.0122, df = 10, p-value = 0.7243
```

Given the low p-value, we reject the hypothesis that there is no difference between men's and women's party support. Gender seems to map onto party preferences.

A very nice way to create cross-tabulations is to go back to the 'modelsummary' package:

```
library(modelsummary)
datasummary_crosstab(female ~ vote, data = D, statistic = 1 ~ 1 + N + Percent("col"))
```

female		Melenchon	Jadot	Macron	Peceresse	Le Pen	Zemmour	All
0	N	180	28	149	25	110	39	805
	% col	46.8	40.0	47.0	50.0	43.1	55.7	46.7
1	N	204	42	168	25	145	31	916
	% col	53.0	60.0	53.0	50.0	56.9	44.3	53.2
2	N	1	0	0	0	0	0	1
	% col	0.3	0.0	0.0	0.0	0.0	0.0	0.1
All	N	385	70	317	50	255	70	1722
	% col	100.0	100.0	100.0	100.0	100.0	100.0	100.0

This shows the same results as earlier, but in a much nicer format!