Describing and summarizing data

Abhijit Dasgupta

Fall, 2019

Where we've been

1. Understand what tidy data is
2. Manipulate data to make it tidy (tidyr, dplyr)
3. Transform particular variables
4. Write basic functions
5. High-throughput analyses
    - Lists of data sets
    - `map` to apply similar processes to each data set
    - for-loops to repeat same recipe on multiple data sets or objects

Where we're going

1. Creating data summaries
2. Basic statistical comparisons between groups
3. Creating tables
    - Table 1
    - Tables for analytic results

The basic assumption we'll make is that we will start with a tidy data set.

Statistical summaries

Univariate summaries

Single summaries

- Mean (`mean`)
- Variance(`var`)
- Standard deviation (`sd`)
- Count (`nrow` or `dplyr::n` or `dplyr::n_distinct`)
- Median ('median')
- Inter-quartile range (`IQR`)
- Mean absolute deviation (`mad`)
- Minimum (`min`) and Maximum (`max`)

Multiple summaries

- Quantiles (`quantile`)
- Range (`range`)

Summarizing the breast cancer expression dataset

Mean

```
brca <- rio::import('data/BreastCancer_Expression.csv
brca %>%
  summarize_at(vars(starts_with('NP')),
               mean, na.rm=T)
```

```
#>    NP_958782 NP_958785 NP_958786 NP_000436 NP_9587
#>  1 0.3202321 0.3269153 0.3264254 0.3236833 0.32708
#>    NP_958784  NP_112598 NP_001611
#>  1 0.3259995 -0.3074577 0.4578748
```

Median

```r
brca %>%
  summarize_at(vars(starts_with('NP')),
                            median, na.rm=T)
```

```
#>    NP_958782 NP_958785 NP_958786 NP_000436 NP_9587
#>  1 0.3236627 0.3269726 0.3269726 0.3302826 0.32697
#>    NP_958784  NP_112598 NP_001611
#>  1 0.3269726 -0.6021319 0.6948104
```

Standard deviation

```
brca %>%
  summarize_at(vars(starts_with('NP')),
               sd, na.rm=T)
```

```
#>    NP_958782 NP_958785 NP_958786 NP_000436 NP_9587
#>  1 0.9767777 0.9800721 0.9799358 0.9784656 0.98060
#>    NP_958784 NP_112598 NP_001611
#>  1 0.9807512  2.024663  1.496951
```

Multiple summaries together

```
brca %>%
  summarize_at(vars(starts_with('NP')),
               c(mean,
               median,
               sd), na.rm=T)
```

```
#>    NP_958782_fn1 NP_958785_fn1 NP_958786_fn1 NP_00
#>  1     0.3202321     0.3269153     0.3264254     0
#>    NP_958780_fn1 NP_958783_fn1 NP_958784_fn1 NP_11
#>  1     0.3263382     0.3259212     0.3259995    -0
#>    NP_958782_fn2 NP_958785_fn2 NP_958786_fn2 NP_00
#>  1     0.3236627     0.3269726     0.3269726     0
#>    NP_958780_fn2 NP_958783_fn2 NP_958784_fn2 NP_11
#>  1     0.3269726     0.3269726     0.3269726    -0
#>    NP_958782_fn3 NP_958785_fn3 NP_958786_fn3 NP_00
#>  1     0.9767777     0.9800721     0.9799358     0
#>    NP_958780_fn3 NP_958783_fn3 NP_958784_fn3 NP_11
#>  1     0.9796277     0.9806739     0.9807512
```

Multiple summaries together

```
brca %>%
  summarize_at(-1, # got tired of typing
               c('Mean'=mean,
                 'Median' = median,
                 'SD'=sd), na.rm=T)
```

```
#>    NP_958782_Mean NP_958785_Mean NP_958786_Mean NP
#>  1      0.3202321      0.3269153      0.3264254
#>    NP_958781_Mean NP_958780_Mean NP_958783_Mean NP
#>  1      0.3270832      0.3263382      0.3259212
#>    NP_112598_Mean NP_001611_Mean NP_958782_Median
#>  1     -0.3074577      0.4578748        0.3236627
#>    NP_958786_Median NP_000436_Median NP_958781_Med
#>  1        0.3269726        0.3302826         0.3269
#>    NP_958783_Median NP_958784_Median NP_112598_Med
#>  1        0.3269726        0.3269726        -0.6021
#>    NP_958782_SD NP_958785_SD NP_958786_SD NP_00043
#>  1    0.9767777    0.9800721    0.9799358    0.978
#>    NP_958780_SD NP_958783_SD NP_958784_SD NP_11259
#>  1    0.9796277    0.9806739    0.9807512      2.02
```

Multiple summaries together

```r
brca %>%
  summarize_at(-1,
               c('Mean' = mean,
                 'Median' = median,
                 'SD' = sd), na.rm=T) %>%
  tidyr::gather(variable, value) %>%
  separate(variable,
           c("Type",'ID','Statistic'), sep='_') %>%
  spread(Statistic, value) %>%
  unite(ID, c('Type','ID'), sep='_')
```

```
#>            ID       Mean     Median        SD
#> 1  NP_000436  0.3236833  0.3302826 0.9784656
#> 2  NP_001611  0.4578748  0.6948104 1.4969506
#> 3  NP_112598 -0.3074577 -0.6021319 2.0246634
#> 4  NP_958780  0.3263382  0.3269726 0.9796277
#> 5  NP_958781  0.3270832  0.3269726 0.9806001
#> 6  NP_958782  0.3202321  0.3236627 0.9767777
#> 7  NP_958783  0.3259212  0.3269726 0.9806739
#> 8  NP_958784  0.3259995  0.3269726 0.9807512
#> 9  NP_958785  0.3269153  0.3269726 0.9800721
#> 10 NP_958786  0.3264254  0.3269726 0.9799358
```

The highlighted part is to format the output

Data set summary

There is a function `summary` that will give you summaries of all the variables. It's nice for looking at the data, but the output format isn't very good for further manipulation

```
summary(brca[,-1])
```

```
#>     NP_958782          NP_958785          NP_958786
#>  Min.   :-1.9478   Min.   :-1.9527   Min.   :-1.9
#>  1st Qu.:-0.4549   1st Qu.:-0.4421   1st Qu.:-0.4
#>  Median : 0.3237   Median : 0.3270   Median : 0.3
#>  Mean   : 0.3202   Mean   : 0.3269   Mean   : 0.3
#>  3rd Qu.: 0.9181   3rd Qu.: 0.9238   3rd Qu.: 0.9
#>  Max.   : 2.7651   Max.   : 2.7797   Max.   : 2.7
#>     NP_958781          NP_958780          NP_958783
#>  Min.   :-1.9576   Min.   :-1.9552   Min.   :-1.9
#>  1st Qu.:-0.4440   1st Qu.:-0.4458   1st Qu.:-0.4
#>  Median : 0.3270   Median : 0.3270   Median : 0.3
#>  Mean   : 0.3271   Mean   : 0.3263   Mean   : 0.3
#>  3rd Qu.: 0.9277   3rd Qu.: 0.9238   3rd Qu.: 0.9
#>  Max.   : 2.7870   Max.   : 2.7797   Max.   : 2.7
#>     NP_112598          NP_001611
#>  Min.   :-4.9527   Min.   :-2.5751
#>  1st Qu.:-1.6741   1st Qu.:-0.5216
#>  Median :-0.6021   Median : 0.6948
#>  Mean   :-0.3075   Mean   : 0.4579
#>  3rd Qu.: 0.8696   3rd Qu.: 1.4394
#>  Max.   : 4.9557   Max.   : 3.4365
```

Maybe an easier way?

The `tableone` package

The `tableone` package is meant to create, you guessed it, Table 1.

It is quite a convenient package for most purposes and saves gobs of time

The `tableone` package

```r
library(tableone)
tab1 <- CreateTableOne(data=brca[,-1])
tab1
```

```
#>
#>                             Overall
#>    n                           83
#>    NP_958782 (mean (SD))  0.32 (0.98)
#>    NP_958785 (mean (SD))  0.33 (0.98)
#>    NP_958786 (mean (SD))  0.33 (0.98)
#>    NP_000436 (mean (SD))  0.32 (0.98)
#>    NP_958781 (mean (SD))  0.33 (0.98)
#>    NP_958780 (mean (SD))  0.33 (0.98)
#>    NP_958783 (mean (SD))  0.33 (0.98)
#>    NP_958784 (mean (SD))  0.33 (0.98)
#>    NP_112598 (mean (SD)) -0.31 (2.02)
#>    NP_001611 (mean (SD))  0.46 (1.50)
```

The `tableone` package

```r
library(tableone)
tab1 <- CreateTableOne(data = brca[-1])
print(tab1, nonnormal = names(brca)[-1])
```

You have to give the variable names of those you think are non-normally distributed and need to be summarized by the median

```
#>
#>                                  Overall
#>    n                               83
#>    NP_958782 (median [IQR])  0.32 [-0.45, 0.92]
#>    NP_958785 (median [IQR])  0.33 [-0.44, 0.92]
#>    NP_958786 (median [IQR])  0.33 [-0.44, 0.92]
#>    NP_000436 (median [IQR])  0.33 [-0.44, 0.92]
#>    NP_958781 (median [IQR])  0.33 [-0.44, 0.93]
#>    NP_958780 (median [IQR])  0.33 [-0.45, 0.92]
#>    NP_958783 (median [IQR])  0.33 [-0.44, 0.92]
#>    NP_958784 (median [IQR])  0.33 [-0.44, 0.92]
#>    NP_112598 (median [IQR]) -0.60 [-1.67, 0.87]
#>    NP_001611 (median [IQR])  0.69 [-0.52, 1.44]
```

The `tableone` package

```
library(tableone)
tab1 <- CreateTableOne(data = brca[-1])
kableone(print(tab1, nonnormal = names(brca)[-1]),
         format='html')
```

	Overall
n	83
NP_958782 (median [IQR])	0.32 [-0.45, 0.92]
NP_958785 (median [IQR])	0.33 [-0.44, 0.92]
NP_958786 (median [IQR])	0.33 [-0.44, 0.92]
NP_000436 (median [IQR])	0.33 [-0.44, 0.92]
NP_958781 (median [IQR])	0.33 [-0.44, 0.93]
NP_958780 (median [IQR])	0.33 [-0.45, 0.92]
NP_958783 (median [IQR])	0.33 [-0.44, 0.92]
NP_958784 (median [IQR])	0.33 [-0.44, 0.92]
NP_112598 (median [IQR])	-0.60 [-1.67, 0.87]
NP_001611 (median [IQR])	0.69 [-0.52, 1.44]

Mixed data

Let's first put the expression and clinical data together

```
library(rio)
brca1 <- import('data/clinical_data_breast_cancer_hw.csv')
brca2 <- import('data/BreastCancer_Expression.csv')
brca <- left_join(brca1, brca2, by=c('Complete.TCGA.ID' = 'TCGA_ID')) %>%
  mutate(Age.at.Initial.Pathologic.Diagnosis =
           as.numeric(Age.at.Initial.Pathologic.Diagnosis)) %>%
  mutate(ER.Status = ifelse(ER.Status %in% c('Positive','Negative'),
                            ER.Status, NA))
```

```
#>  Warning: NAs introduced by coercion
```

```
summary(brca)
```

```
#>   Complete.TCGA.ID      Gender          Age.at.Initial.Pathologic.Diagnosis
#>   Length:108          Length:108         Min.   :30.00
#>   Class :character    Class :character   1st Qu.:49.00
#>   Mode  :character    Mode  :character   Median :58.00
#>                                          Mean   :58.72
#>                                          3rd Qu.:66.50
#>                                          Max.   :88.00
#>                                          NA's   :1
#>    ER.Status           PR.Status         HER2.Final.Status
#>   Length:108          Length:108         Length:108
#>   Class :character    Class :character   Class :character
#>   Mode  :character    Mode  :character   Mode  :character
#>
#>
#>
#>
```

Let's first put the expression and clinical data together

```
library(rio)
brca1 <- import('data/clinical_data_breast_cancer_hw.csv')
brca2 <- import('data/BreastCancer_Expression.csv')
brca <- left_join(brca1, brca2, by=c('Complete.TCGA.ID' = 'TCGA_ID')) %>%
  mutate(Age.at.Initial.Pathologic.Diagnosis =
           as.numeric(Age.at.Initial.Pathologic.Diagnosis)) %>%
  mutate(ER.Status = ifelse(ER.Status %in% c('Positive','Negative'),
                            ER.Status, NA),
         HER2.Final.Status = ifelse(HER2.Final.Status=='Equivocal',
                                    NA, HER2.Final.Status)) %>%
  mutate_if(is.character, as.factor) %>%
  mutate(Complete.TCGA.ID = as.character(Complete.TCGA.ID))
```

```
#>  Warning: NAs introduced by coercion
```

```
str(brca)
```

```
#>  'data.frame':    108 obs. of  23 variables:
#>   $ Complete.TCGA.ID                 : chr  "TCGA-A2-A0T2" "TCGA-A2-A0CM" "TCGA-BH-A18V" "TCGA-BH-A18Q" ...
#>   $ Gender                           : Factor w/ 2 levels "FEMALE","MALE": 1 1 1 1 1 1 1 1 1 1 ...
#>   $ Age.at.Initial.Pathologic.Diagnosis: num  66 40 48 56 38 57 74 60 61 NA ...
#>   $ ER.Status                        : Factor w/ 2 levels "Negative","Positive": 1 1 1 1 1 1 1 1 1 1 ...
#>   $ PR.Status                        : Factor w/ 2 levels "Negative","Positive": 1 1 1 1 1 1 1 1 1 1 ...
#>   $ HER2.Final.Status                : Factor w/ 2 levels "Negative","Positive": 1 1 1 1 1 1 1 1 1 1 ...
#>   $ Tumor                            : Factor w/ 4 levels "T1","T2","T3",..: 3 2 2 2 3 2 3 2 2 2 ...
#>   $ Node                             : Factor w/ 4 levels "N0","N1","N2",..: 4 1 2 2 4 1 1 1 1 1 ...
#>   $ Metastasis                       : Factor w/ 2 levels "M0","M1": 2 1 1 1 1 1 1 1 1 1 ...
#>   $ AJCC.Stage                       : Factor w/ 11 levels "Stage I","Stage IA",..: 11 5 6 6 10 5 6 5 5 5 ..
#>   $ Vital.Status                     : Factor w/ 2 levels "DECEASED","LIVING": 1 1 1 1 2 2 2 2 2 2 ...
```

Identify which variables are categorical (factors) and which are continuous (numeric)

```
catvars <- brca %>% select_if(is.factor) %>% names()
ctsvars <- brca %>% select_if(is.numeric) %>% names()
```

```
CreateCatTable(vars = catvars, data = brca)
```

```
#>
#>                                       Overall
#>    n                                  108
#>    Gender = MALE (%)                   2 ( 1.9)
#>    ER.Status = Positive (%)           69 (64.5)
#>    PR.Status = Positive (%)           55 (50.9)
#>    HER2.Final.Status = Positive (%) 28 (26.2)
#>    Tumor (%)
#>       T1                              16 (14.8)
#>       T2                              67 (62.0)
#>       T3                              19 (17.6)
#>       T4                               6 ( 5.6)
#>    Node (%)
#>       N0                              54 (50.0)
#>       N1                              30 (27.8)
#>       N2                              15 (13.9)
#>       N3                               9 ( 8.3)
#>    Metastasis = M1 (%)                 2 ( 1.9)
#>    AJCC.Stage (%)
#>       Stage I                          3 ( 2.8)
#>       Stage IA                         7 ( 6.5)
#>       Stage IB                         2 ( 1.9)
#>       Stage II                        11 (10.2)
#>       Stage IIA                       32 (29.6)
#>       Stage IIB                       23 (21.3)
#>       Stage III                        4 ( 3.7)
#>       Stage IIIA                      12 (11.1)
#>       Stage IIIB                       6 ( 5.6)
#>       Stage IIIC                       6 ( 5.6)
#>       Stage IV                         2 ( 1.9)
#>    Vital.Status = LIVING (%)          97 (89.8)
```

```
CreateContTable(vars = ctsvars, data = brca)
```

```
#>
#>
#>    n
#>    Age.at.Initial.Pathologic.Diagnosis (mean (SD))
#>    Days.to.Date.of.Last.Contact (mean (SD))
#>    Days.to.date.of.Death (mean (SD))
#>    NP_958782 (mean (SD))
#>    NP_958785 (mean (SD))
#>    NP_958786 (mean (SD))
#>    NP_000436 (mean (SD))
#>    NP_958781 (mean (SD))
#>    NP_958780 (mean (SD))
#>    NP_958783 (mean (SD))
#>    NP_958784 (mean (SD))
#>    NP_112598 (mean (SD))
#>    NP_001611 (mean (SD))
```

```r
brca <- brca %>%
  rename(
    'Age'='Age.at.Initial.Pathologic.Diagnosis',
    'Last.Contact' = 'Days.to.Date.of.Last.Contact',
    'Death' = 'Days.to.date.of.Death'
  )
ctsvars <- brca %>% select_if(is.numeric) %>% names()
CreateContTable(vars = ctsvars, data = brca)
```

```
#>
#>                              Overall
#>    n                         108
#>    Age (mean (SD))            58.72 (13.21)
#>    Last.Contact (mean (SD))  806.37 (667.70)
#>    Death (mean (SD))         1254.45 (678.05)
#>    NP_958782 (mean (SD))        0.32 (0.99)
#>    NP_958785 (mean (SD))        0.33 (1.00)
#>    NP_958786 (mean (SD))        0.33 (1.00)
#>    NP_000436 (mean (SD))        0.32 (0.99)
#>    NP_958781 (mean (SD))        0.33 (1.00)
#>    NP_958780 (mean (SD))        0.33 (1.00)
#>    NP_958783 (mean (SD))        0.33 (1.00)
#>    NP_958784 (mean (SD))        0.33 (1.00)
#>    NP_112598 (mean (SD))       -0.30 (2.06)
#>    NP_001611 (mean (SD))        0.38 (1.46)
```

Putting it together

```
CreateTableOne(vars = c(catvars, ctsvars),
               data = brca)
```

```
#> 
#>                                          Overall
#>    n                                       108
#>    Gender = MALE (%)                         2 ( 1.9)
#>    ER.Status = Positive (%)                 69 (64.5)
#>    PR.Status = Positive (%)                 55 (50.9)
#>    HER2.Final.Status = Positive (%)         28 (26.2)
#>    Tumor (%)
#>       T1                                    16 (14.8)
#>       T2                                    67 (62.0)
#>       T3                                    19 (17.6)
#>       T4                                     6 ( 5.6)
#>    Node (%)
#>       N0                                    54 (50.0)
#>       N1                                    30 (27.8)
#>       N2                                    15 (13.9)
#>       N3                                     9 ( 8.3)
#>    Metastasis = M1 (%)                       2 ( 1.9)
#>    AJCC.Stage (%)
#>       Stage I                                3 ( 2.8)
#>       Stage IA                               7 ( 6.5)
#>       Stage IB                               2 ( 1.9)
#>       Stage II                              11 (10.2)
#>       Stage IIA                             32 (29.6)
#>       Stage IIB                             23 (21.3)
#>       Stage III                              4 ( 3.7)
#>       Stage IIIA                            12 (11.1)
#>       Stage IIIB                             6 ( 5.6)
```

Putting it together

```r
CreateTableOne(data = brca[,-1])
```

```
#> 
#>                                        Overall
#>     n                                      108
#>     Gender = MALE (%)                        2 ( 1.9)
#>     Age (mean (SD))                      58.72 (13.21
#>     ER.Status = Positive (%)                69 (64.5)
#>     PR.Status = Positive (%)                55 (50.9)
#>     HER2.Final.Status = Positive (%)        28 (26.2)
#>     Tumor (%)                                  
#>        T1                                   16 (14.8)
#>        T2                                   67 (62.0)
#>        T3                                   19 (17.6)
#>        T4                                    6 ( 5.6)
#>     Node (%)                                   
#>        N0                                   54 (50.0)
#>        N1                                   30 (27.8)
#>        N2                                   15 (13.9)
#>        N3                                    9 ( 8.3)
#>     Metastasis = M1 (%)                      2 ( 1.9)
#>     AJCC.Stage (%)                             
#>        Stage I                               3 ( 2.8)
#>        Stage IA                              7 ( 6.5)
#>        Stage IB                              2 ( 1.9)
#>        Stage II                             11 (10.2)
#>        Stage IIA                            32 (29.6)
#>        Stage IIB                            23 (21.3)
#>        Stage III                             4 ( 3.7)
#>        Stage IIIA                           12 (11.1)
```

Grouped summaries

```
brca %>%
  group_by(ER.Status) %>%
  summarize_at(vars(starts_with('NP')),
               mean)
```

There are missing values now, so we have to use `na.rm=T`.

```
#>  # A tibble: 3 x 11
#>    ER.Status NP_958782 NP_958785 NP_958786 NP_0004
#>    <fct>         <dbl>     <dbl>     <dbl>     <db
#>  1 Negative         NA        NA        NA
#>  2 Positive         NA        NA        NA
#>  3 <NA>             NA        NA        NA
#>  # … with 4 more variables: NP_958783 <dbl>, NP_95
#>  #   NP_112598 <dbl>, NP_001611 <dbl>
```

```
brca %>%
  group_by(ER.Status) %>%
  summarize_at(vars(starts_with('NP')),
               mean, na.rm=T)
```

We still have a row for the missing values of ER.Status

```
#>  # A tibble: 3 x 11
#>    ER.Status NP_958782 NP_958785 NP_958786 NP_0004
#>    <fct>         <dbl>     <dbl>     <dbl>     <db
#>  1 Negative      0.429     0.438     0.439     0.4
#>  2 Positive      0.267     0.273     0.272     0.2
#>  3 <NA>        NaN       NaN       NaN       NaN
#>  # … with 4 more variables: NP_958783 <dbl>, NP_95
#>  #   NP_112598 <dbl>, NP_001611 <dbl>
```

```r
brca %>%
  filter(!is.na(ER.Status)) %>%
  group_by(ER.Status) %>%
  summarize_at(vars(starts_with('NP')),
               mean, na.rm=T)
```

```
#>  # A tibble: 2 x 11
#>    ER.Status NP_958782 NP_958785 NP_958786 NP_0004
#>    <fct>         <dbl>     <dbl>     <dbl>     <db
#>  1 Negative      0.429     0.438     0.439     0.4
#>  2 Positive      0.267     0.273     0.272     0.2
#>  # … with 4 more variables: NP_958783 <dbl>, NP_95
#>  #   NP_112598 <dbl>, NP_001611 <dbl>
```

How about reversing the rows and columns for readability

```
brca %>%
  filter(!is.na(ER.Status)) %>%
  group_by(ER.Status) %>%
  summarize_at(vars(starts_with('NP')),
               mean, na.rm=T) %>%
  tidyr::gather(ID, value, -ER.Status) %>%
  spread(ER.Status, value)
```

```
#>  # A tibble: 10 x 3
#>     ID        Negative Positive
#>     <chr>        <dbl>    <dbl>
#>   1 NP_000436    0.432    0.271
#>   2 NP_001611   -0.566    0.840
#>   3 NP_112598   -0.197   -0.357
#>   4 NP_958780    0.436    0.273
#>   5 NP_958781    0.436    0.274
#>   6 NP_958782    0.429    0.267
#>   7 NP_958783    0.436    0.272
#>   8 NP_958784    0.436    0.273
#>   9 NP_958785    0.438    0.273
#>  10 NP_958786    0.439    0.272
```

Using `tableone`

```
CreateTableOne(
  data = brca %>% filter(!is.na(ER.Status)),
  vars = brca %>%
    select(starts_with('NP')) %>%
    names(),
  strata = 'ER.Status',
  test = F)
```

```
#>                          Stratified by ER.Status
#>                           Negative      Positive
#>    n                         38            69
#>    NP_958782 (mean (SD))  0.43 (1.13)  0.27 (0.93)
#>    NP_958785 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_958786 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_000436 (mean (SD))  0.43 (1.14)  0.27 (0.93)
#>    NP_958781 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_958780 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_958783 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_958784 (mean (SD))  0.44 (1.14)  0.27 (0.93)
#>    NP_112598 (mean (SD)) -0.20 (2.28) -0.36 (1.97)
#>    NP_001611 (mean (SD)) -0.57 (1.54)  0.84 (1.19)
```

Comparing two groups

The t-test

The t-test compares whether the mean of a variable differs between two groups.

It does assume the normal distribution for the data, but is robust to deviations from normality

Do **not** test for normality before doing the t-test. It isn't necessary and screws up your error rates

The t-test

In R, there is a convenient function `t.test`

```
t.test(NP_958782 ~ ER.Status, data = brca)
```

```
#> 
#>     Welch Two Sample t-test
#> 
#> data:  NP_958782 by ER.Status
#> t = 0.63522, df = 41.807, p-value = 0.5287
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#>  -0.3523151  0.6759226
#> sample estimates:
#> mean in group Negative mean in group Positive 
#>              0.4292798              0.2674761
```

Read the code as

"Do a t-test to see if (the mean of) `NP_958782` differs by `ER.Status`, where both are taken from the data set `brca`"

You can read the ~ as "by", as in "t-test of NP_958782 by ER.Status"

The t-test

The packge `broom` provides a function `tidy` that makes the results of these statistical tests tidy.

```
t.test(NP_958782 ~ ER.Status, data=brca) %>%
  broom::tidy()
```

```
#>  # A tibble: 1 x 10
#>    estimate estimate1 estimate2 statistic p.value parameter conf.low
#>       <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>
#>  1    0.162     0.429     0.267     0.635   0.529      41.8   -0.352
#>  # … with 3 more variables: conf.high <dbl>, method <chr>,
#>  #   alternative <chr>
```

```
#>
#>      Welch Two Sample t-test
#>
#>  data:  NP_958782 by ER.Status
#>  t = 0.63522, df = 41.807, p-value = 0.5287
#>  alternative hypothesis: true difference in means is not equal to 0
#>  95 percent confidence interval:
#>   -0.3523151  0.6759226
#>  sample estimates:
#>  mean in group Negative mean in group Positive
#>               0.4292798              0.2674761
```

Using `broom`

The fact that `broom::tidy` makes the results of tests into tibbles is in fact extremely useful in high-throughput work

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status)
```

```
#>      ER.Status   protein  expression
#>  1    Negative NP_958782          NA
#>  2    Negative NP_958782  0.68340354
#>  3    Negative NP_958782          NA
#>  4    Negative NP_958782  0.19534065
#>  5    Negative NP_958782          NA
#>  6    Negative NP_958782 -1.12317308
#>  7    Negative NP_958782  0.53859578
#>  8    Negative NP_958782          NA
#>  9    Negative NP_958782  0.83113175
#>  10   Negative NP_958782  0.65584968
#>  11   Negative NP_958782  0.10749090
#>  12   Negative NP_958782 -0.39855983
#>  13   Negative NP_958782 -0.10667998
#>  14   Negative NP_958782 -1.94779243
#>  15   Negative NP_958782  0.32366271
#>  16   Negative NP_958782  2.45513793
#>  17   Negative NP_958782 -0.03322133
#>  18   Negative NP_958782          NA
#>  19   Negative NP_958782  0.35053566
#>  20   Negative NP_958782  0.67390470
#>  21   Negative NP_958782  2.60994298
#>  22   Negative NP_958782  2.70725015
#>  23   Negative NP_958782  0.14018179
```

Using broom

The fact that `broom::tidy` makes the results of tests into tibbles is in fact extremely useful in high-throughput work

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status) %>%
  group_split(protein)
```

```
#>  [[1]]
#>  # A tibble: 108 x 3
#>     ER.Status protein   expression
#>     <fct>     <chr>          <dbl>
#>   1 Negative  NP_000436     NA
#>   2 Negative  NP_000436      0.687
#>   3 Negative  NP_000436     NA
#>   4 Negative  NP_000436      0.205
#>   5 Negative  NP_000436     NA
#>   6 Negative  NP_000436     -1.13
#>   7 Negative  NP_000436      0.535
#>   8 Negative  NP_000436     NA
#>   9 Negative  NP_000436      0.837
#>  10 Negative  NP_000436      0.656
#>  # … with 98 more rows
#>
#>  [[2]]
#>  # A tibble: 108 x 3
#>     ER.Status protein   expression
#>     <fct>     <chr>          <dbl>
#>   1 Negative  NP_001611     NA
#>   2 Negative  NP_001611     -0.984
#>   3 Negative  NP_001611     NA
#>   4 Negative  NP_001611     -0.517
```

Using broom

The fact that `broom::tidy` makes the results of tests into tibbles is in fact extremely useful in high-throughput work

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status) %>%
  split(.$protein) %>%
  map(~broom::tidy(t.test(expression ~ ER.Status,
                          data=.)))
```

```
#> $NP_000436
#> # A tibble: 1 x 10
#>   estimate estimate1 estimate2 statistic p.value
#>      <dbl>     <dbl>     <dbl>     <dbl>   <dbl>
#> 1    0.161     0.432     0.271     0.628   0.534
#> # … with 3 more variables: conf.high <dbl>, metho
#> #   alternative <chr>
#>
#> $NP_001611
#> # A tibble: 1 x 10
#>   estimate estimate1 estimate2 statistic p.value
#>      <dbl>     <dbl>     <dbl>     <dbl>   <dbl>
#> 1    -1.41    -0.566     0.840     -4.10 1.99e-4
#> # … with 3 more variables: conf.high <dbl>, metho
#> #   alternative <chr>
#>
#> $NP_112598
#> # A tibble: 1 x 10
#>   estimate estimate1 estimate2 statistic p.value
#>      <dbl>     <dbl>     <dbl>     <dbl>   <dbl>
#> 1    0.160    -0.197    -0.357     0.306   0.761
#> # … with 3 more variables: conf.high <dbl>, metho
#> #   alternative <chr>
#>
```

Using broom

The fact that `broom::tidy` makes the results of tests into tibbles is in fact extremely useful in high-throughput work

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status) %>%
  split(.$protein) %>%
  map(~broom::tidy(t.test(expression ~ ER.Status,
                          data=.))) %>%
  bind_rows(.id='Protein')
```

```
#>  # A tibble: 10 x 11
#>     Protein estimate estimate1 estimate2 statistic
#>     <chr>      <dbl>     <dbl>     <dbl>     <dbl>
#>   1 NP_000…    0.161     0.432     0.271     0.628
#>   2 NP_001…   -1.41     -0.566     0.840    -4.10
#>   3 NP_112…    0.160    -0.197    -0.357     0.306
#>   4 NP_958…    0.163     0.436     0.273     0.637
#>   5 NP_958…    0.162     0.436     0.274     0.633
#>   6 NP_958…    0.162     0.429     0.267     0.635
#>   7 NP_958…    0.164     0.436     0.272     0.639
#>   8 NP_958…    0.164     0.436     0.273     0.639
#>   9 NP_958…    0.165     0.438     0.273     0.642
#>  10 NP_958…    0.166     0.439     0.272     0.649
#>  # … with 4 more variables: conf.low <dbl>, conf.h
#>  #   alternative <chr>
```

Using broom

The fact that `broom::tidy` makes the results of tests into tibbles is in fact extremely useful in high-throughput work

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status) %>%
  split(.$protein) %>%
  map(~broom::tidy(t.test(expression ~ ER.Status,
                          data=.))) %>%
  bind_rows(.id='Protein') %>%
  select(Protein, estimate, p.value, conf.low, conf.h
```

```
#>  # A tibble: 10 x 5
#>     Protein   estimate  p.value conf.low conf.high
#>     <chr>        <dbl>    <dbl>    <dbl>     <dbl>
#>   1 NP_000436    0.161 0.534      -0.356     0.677
#>   2 NP_001611   -1.41  0.000199   -2.10     -0.712
#>   3 NP_112598    0.160 0.761      -0.892     1.21
#>   4 NP_958780    0.163 0.528      -0.354     0.680
#>   5 NP_958781    0.162 0.530      -0.356     0.680
#>   6 NP_958782    0.162 0.529      -0.352     0.676
#>   7 NP_958783    0.164 0.527      -0.354     0.681
#>   8 NP_958784    0.164 0.527      -0.354     0.681
#>   9 NP_958785    0.165 0.524      -0.353     0.682
#>  10 NP_958786    0.166 0.520      -0.351     0.684
```

Back to testing

Wilcoxon test, nonparametric t-test

```
wilcox.test(NP_958782 ~ ER.Status, data=brca) %>%
  broom::tidy()
```

```
#>  # A tibble: 1 x 4
#>    statistic p.value method                                        alternative
#>        <dbl>   <dbl> <chr>                                         <chr>
#>  1       755   0.590 Wilcoxon rank sum test with continuity cor… two.sided
```

```
#>
#>     Wilcoxon rank sum test with continuity correction
#>
#>  data:  NP_958782 by ER.Status
#>  W = 755, p-value = 0.5897
#>  alternative hypothesis: true location shift is not equal to 0
```

Wilcoxon test

```
brca %>%
  select(ER.Status, starts_with('NP')) %>%
  tidyr::gather(protein,expression, -ER.Status) %>%
  split(.$protein) %>%
  map(~broom::tidy(wilcox.test(expression ~ ER.Status
                        data=.))) %>%
  bind_rows(.id='Protein') %>%
  select(Protein, p.value)
```

```
#>  # A tibble: 10 x 2
#>     Protein     p.value
#>     <chr>         <dbl>
#>   1 NP_000436 0.583
#>   2 NP_001611 0.0000928
#>   3 NP_112598 0.939
#>   4 NP_958780 0.583
#>   5 NP_958781 0.576
#>   6 NP_958782 0.590
#>   7 NP_958783 0.583
#>   8 NP_958784 0.576
#>   9 NP_958785 0.576
#>  10 NP_958786 0.576
```

Using `tableone`

```
CreateTableOne(
  data = brca %>% filter(!is.na(ER.Status)),
  vars = brca %>%
    select(starts_with('NP')) %>%
    names(),
  strata = 'ER.Status',
  test = T,
  testNormal = t.test
)
```

```
#>                          Stratified by ER.Status
#>                           Negative     Positive     p      test
#>    n                         38           69
#>    NP_958782 (mean (SD))  0.43 (1.13)  0.27 (0.93)  0.498
#>    NP_958785 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.492
#>    NP_958786 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.487
#>    NP_000436 (mean (SD))  0.43 (1.14)  0.27 (0.93)  0.502
#>    NP_958781 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.499
#>    NP_958780 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.496
#>    NP_958783 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.495
#>    NP_958784 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.495
#>    NP_112598 (mean (SD)) -0.20 (2.28) -0.36 (1.97)  0.748
#>    NP_001611 (mean (SD)) -0.57 (1.54)  0.84 (1.19) <0.001
```

This is not quite the same results as before

Using tableone

```
CreateTableOne(
  data = brca %>% filter(!is.na(ER.Status)),
  vars = brca %>%
    select(starts_with('NP')) %>%
    names(),
  strata = 'ER.Status',
  test = T,
  testNormal = t.test,
  argsNormal = list(var.equal=F)
)
```

```
#>                            Stratified by ER.Status
#>                             Negative      Positive      p      test
#>    n                           38            69
#>    NP_958782 (mean (SD))  0.43 (1.13)  0.27 (0.93)  0.529
#>    NP_958785 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.524
#>    NP_958786 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.520
#>    NP_000436 (mean (SD))  0.43 (1.14)  0.27 (0.93)  0.534
#>    NP_958781 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.530
#>    NP_958780 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.528
#>    NP_958783 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.527
#>    NP_958784 (mean (SD))  0.44 (1.14)  0.27 (0.93)  0.527
#>    NP_112598 (mean (SD)) -0.20 (2.28) -0.36 (1.97)  0.761
#>    NP_001611 (mean (SD)) -0.57 (1.54)  0.84 (1.19) <0.001
```

Tests for discrete data

Testing whether the distribution of a categorical variable differs by levels of another categorical variable can be done using either the Chi-square test (`chisq.test`) or the Fisher's test (`fisher.test`). Both require you to create a 2x2 table first.

```
fisher.test(table(brca$Tumor, brca$ER.Status))
```

```
#> 
#>      Fisher's Exact Test for Count Data
#> 
#>  data:  table(brca$Tumor, brca$ER.Status)
#>  p-value = 0.6003
#>  alternative hypothesis: two.sided
```

Tests for discrete data

Testing whether the distribution of a categorical variable differs by levels of another categorical variable can be done using either the Chi-square test (`chisq.test`) or the Fisher's test (`fisher.test`). Both require you to create a 2x2 table first.

```
chisq.test(table(brca$Tumor, brca$ER.Status))
```

```
#>  Warning in chisq.test(table(brca$Tumor, brca$ER.Status)): Chi-squared
#>  approximation may be incorrect
```

```
#> 
#>      Pearson's Chi-squared test
#> 
#>  data:  table(brca$Tumor, brca$ER.Status)
#>  X-squared = 2.094, df = 3, p-value = 0.5531
```

Tests for discrete data

We can use `broom::tidy` for either of these

```
chisq.test(table(brca$Tumor, brca$ER.Status)) %>%
  broom::tidy()
```

```
#>  Warning in chisq.test(table(brca$Tumor, brca$ER.Status)): Chi-squared
#>  approximation may be incorrect
```

```
#>  # A tibble: 1 x 4
#>    statistic p.value parameter method
#>        <dbl>   <dbl>     <int> <chr>
#>  1      2.09   0.553         3 Pearson's Chi-squared test
```

Using `tableone`

```
CreateCatTable(vars = c('Tumor','Node','Metastasis'),
               data = filter(brca, !is.na(ER.Status)),
               strata = 'ER.Status',
               test = T) # chisq.test
```

```
#>                       Stratified by ER.Status
#>                        Negative   Positive   p      test
#>    n                   38         69
#>    Tumor (%)                                  0.553
#>       T1                6 (15.8)  10 (14.5)
#>       T2               26 (68.4)  40 (58.0)
#>       T3                5 (13.2)  14 (20.3)
#>       T4                1 ( 2.6)   5 ( 7.2)
#>    Node (%)                                   0.685
#>       N0               22 (57.9)  32 (46.4)
#>       N1                8 (21.1)  21 (30.4)
#>       N2                5 (13.2)  10 (14.5)
#>       N3                3 ( 7.9)   6 ( 8.7)
#>    Metastasis = M1 (%)  1 ( 2.6)   1 ( 1.4)   1.000
```

Using `tableone`

```
c1 <- CreateCatTable(vars = c('Tumor','Node','Metastasis'),
                     data = filter(brca, !is.na(ER.Status)),
                     strata = 'ER.Status',
                     test = T)
print(c1, exact = c('Tumor','Node','Metastasis')) # fisher.test
```

```
#>                        Stratified by ER.Status
#>                         Negative    Positive    p      test
#>    n                    38          69
#>    Tumor (%)                                     0.600 exact
#>       T1                 6 (15.8)   10 (14.5)
#>       T2                26 (68.4)   40 (58.0)
#>       T3                 5 (13.2)   14 (20.3)
#>       T4                 1 ( 2.6)    5 ( 7.2)
#>    Node (%)                                      0.695 exact
#>       N0                22 (57.9)   32 (46.4)
#>       N1                 8 (21.1)   21 (30.4)
#>       N2                 5 (13.2)   10 (14.5)
#>       N3                 3 ( 7.9)    6 ( 8.7)
#>    Metastasis = M1 (%)   1 ( 2.6)    1 ( 1.4)    1.000 exact
```