Department of Statistics, University of Auckland

Summer Scholarship 2017-2018

Data wrangling tools in Shiny and Gtk

Name: Owen Jin

Supervisor: Chris Wild

Degree: BCom/Sc

Summary

I have created replacement functions for most of the options given in the “VARIABLES” and “DATASET” tabs of iNZight by using packages outside of base R. These packages include dplyr, tidyr, forcats and stringr. My functions essentially replace the old base R functions.

In addition to the old functions, the functions I created also reproduce the code needed to replicate the results of the function

Variables Tab

Convert to Categorical

Function: convertToCat()

Convert several numeric columns into categorical columns. The new column created is a factor.

```
> converted <- convertToCat(iris, vars = c("Petal.Width"))
> head(converted)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Petal.Width.cat Species
1          5.1         3.5          1.4         0.2             0.2  setosa
2          4.9         3.0          1.4         0.2             0.2  setosa
3          4.7         3.2          1.3         0.2             0.2  setosa
4          4.6         3.1          1.5         0.2             0.2  setosa
5          5.0         3.6          1.4         0.2             0.2  setosa
6          5.4         3.9          1.7         0.4             0.4  setosa
> is.factor(converted$Petal.Width.cat)
[1] TRUE
> cat(code(converted))
iris %>% tibble::add_column(Petal.Width.cat = factor(iris$Petal.Width),      .after = "Petal.Width")
```

Categorical Variables -> Reorder Levels

Function: reorderLevels()

Reorder the factor levels of a categorical column either manually or by descending frequency

```
> reordered <- reorderLevels(iris, var = "Species", new_levels = c("versicolor", "virginica", "setosa"))
> head(reordered)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Species.reord
1          5.1         3.5          1.4         0.2  setosa        setosa
2          4.9         3.0          1.4         0.2  setosa        setosa
3          4.7         3.2          1.3         0.2  setosa        setosa
4          4.6         3.1          1.5         0.2  setosa        setosa
5          5.0         3.6          1.4         0.2  setosa        setosa
6          5.4         3.9          1.7         0.4  setosa        setosa
> levels(reordered$Species.reord)
[1] "versicolor" "virginica"  "setosa"
> cat(code(reordered))
iris %>% tibble::add_column(Species.reord = factor(iris$Species,      levels = c("versicolor", "virginica", "setosa")), .a
fter = "Species")
>
```

Categorical Variables -> Collapse Levels

Function: collapseLevels()

Collapse 2 or more levels of a factor in a categorical variable into one factor.

```
> collapsed <- collapseLevels(iris, var = "Species", levels = c("setosa", "virginica"))
> head(collapsed)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species     Species.coll
1          5.1         3.5          1.4         0.2  setosa setosa_virginica
2          4.9         3.0          1.4         0.2  setosa setosa_virginica
3          4.7         3.2          1.3         0.2  setosa setosa_virginica
4          4.6         3.1          1.5         0.2  setosa setosa_virginica
5          5.0         3.6          1.4         0.2  setosa setosa_virginica
6          5.4         3.9          1.7         0.4  setosa setosa_virginica
> unique(collapsed$Species.coll)
[1] setosa_virginica versicolor
Levels: setosa_virginica versicolor
> cat(code(collapsed))
iris %>% tibble::add_column(Species.coll = forcats::fct_collapse(iris$Species,      setosa_virginica = c("setosa",
 "virginica")), .after = "Species")
```

Categorical Variables -> Rename Levels

Function: renameLevels()

Rename the factor levels of a categorical column

```
> renamed <- renameLevels(iris, var = "Species", to_be_renamed =
+ list(set = "setosa", ver = "versicolor"))
> head(renamed)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Species.rename
1          5.1         3.5          1.4         0.2  setosa            set
2          4.9         3.0          1.4         0.2  setosa            set
3          4.7         3.2          1.3         0.2  setosa            set
4          4.6         3.1          1.5         0.2  setosa            set
5          5.0         3.6          1.4         0.2  setosa            set
6          5.4         3.9          1.7         0.4  setosa            set
> unique(renamed$Species.rename)
[1] set       ver       virginica
Levels: set ver virginica
> cat(code(renamed))
iris %>% tibble::add_column(Species.rename = forcats::fct_recode(iris$Species,      set = "setosa", ver = "versico
lor"), .after = "Species")
```

Categorical Variables -> Combine Categorical Levels

Function: combineCatVars()

Concatenate the values of several categorical columns into one value

```
> combined <- combineCatVars(warpbreaks, vars = c("wool", "tension"), sep = "_")
> head(combined)
  breaks wool tension wool_tension
1     26    A       L           A_L
2     30    A       L           A_L
3     54    A       L           A_L
4     25    A       L           A_L
5     70    A       L           A_L
6     52    A       L           A_L
> unique(combined$wool_tension)
[1] "A_L" "A_M" "A_H" "B_L" "B_M" "B_H"
> cat(code(combined))
warpbreaks %>% dplyr::mutate(wool_tension = paste(wool, tension,      sep = "_"))
```

Numeric Variables -> Transform Variables

Function: transformVar()

Transform a numeric column and apply a mathematical function

```
> transformed <- transformVar(iris, var = "Petal.Length", transformation = "log")
> head(transformed)
  Sepal.Length Sepal.Width Petal.Length log.Petal.Length Petal.Width Species
1          5.1         3.5          1.4        0.3364722         0.2  setosa
2          4.9         3.0          1.4        0.3364722         0.2  setosa
3          4.7         3.2          1.3        0.2623643         0.2  setosa
4          4.6         3.1          1.5        0.4054651         0.2  setosa
5          5.0         3.6          1.4        0.3364722         0.2  setosa
6          5.4         3.9          1.7        0.5306283         0.4  setosa
> all(transformed$log.Petal.Length == log(transformed$Petal.Length))
[1] TRUE
> cat(code(transformed))
iris %>% tibble::add_column(log.Petal.Length = log(iris$Petal.Length),      .after = "Petal.Length")
```

Numeric Variables -> Standardize Variables

Function: standardizeVars()

Centre then divide by the standard error of the values in a numeric column

```
> standardized <- standardizeVars(iris, var = c("Sepal.Width", "Petal.Width"))
> head(standardized)
  Sepal.Length Sepal.Width Sepal.Width.std Petal.Length Petal.Width Petal.Width.std Species
1          5.1         3.5      1.01560199          1.4         0.2       -1.311052  setosa
2          4.9         3.0     -0.13153881          1.4         0.2       -1.311052  setosa
3          4.7         3.2      0.32731751          1.3         0.2       -1.311052  setosa
4          4.6         3.1      0.09788935          1.5         0.2       -1.311052  setosa
5          5.0         3.6      1.24503015          1.4         0.2       -1.311052  setosa
6          5.4         3.9      1.93331463          1.7         0.4       -1.048667  setosa
> all(standardized$Sepal.Width.std == scale(standardized$Sepal.Width))
[1] TRUE
> cat(code(standardized))
iris %>% tibble::add_column(Sepal.Width.std = scale(iris$Sepal.Width)[,      1], .after = "Sepal.Width") %>% tibbl
e::add_column(Petal.Width.std = scale(iris$Petal.Width)[,      1], .after = "Petal.Width")
>
```

Numeric Variables -> Rank Numeric Variables

Function: rankVars()

Rank the values of a numeric column in descending order. Ties are broken as such: eg. values = 5, 6, 6, 7 ; rank = 1, 2, 2, 3

```
> ranked <- rankVars(iris, vars = c("Sepal.Length", "Petal.Length"))
> head(ranked)
  Sepal.Length Sepal.Length.rank Sepal.Width Petal.Length Petal.Length.rank Petal.Width Species
1          5.1                33         3.5          1.4                12         0.2  setosa
2          4.9                17         3.0          1.4                12         0.2  setosa
3          4.7                10         3.2          1.3                 5         0.2  setosa
4          4.6                 6         3.1          1.5                25         0.2  setosa
5          5.0                23         3.6          1.4                12         0.2  setosa
6          5.4                47         3.9          1.7                45         0.4  setosa
> cat(code(ranked))
iris %>% tibble::add_column(Sepal.Length.rank = dplyr::min_rank(iris$Sepal.Length),      .after = "Sepal.Length")
%>% tibble::add_column(Petal.Length.rank = dplyr::min_rank(iris$Petal.Length),      .after = "Petal.Length")
```

Rename Variables

Function: renameVars()

Rename the column names

```
> renamed <- renameVars(iris, to_be_renamed_list = list(Species = "Type", Petal.Width = "P.W"))
> head(renamed)
  Sepal.Length Sepal.Width Petal.Length P.W   Type
1          5.1         3.5          1.4 0.2 setosa
2          4.9         3.0          1.4 0.2 setosa
3          4.7         3.2          1.3 0.2 setosa
4          4.6         3.1          1.5 0.2 setosa
5          5.0         3.6          1.4 0.2 setosa
6          5.4         3.9          1.7 0.4 setosa
> cat(code(renamed))
iris %>% dplyr::rename(Type = Species, P.W = Petal.Width)
```

Create New Variables

Function: createNewVar()

Create a new column using an R expression

```
> created <- createNewVar(iris, new_var = "Sepal.Length_less_Sepal.Width",
+  "Sepal.Length - Sepal.Width")
> head(created)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.Length_less_Sepal.Width
1          5.1         3.5          1.4         0.2  setosa                            1.6
2          4.9         3.0          1.4         0.2  setosa                            1.9
3          4.7         3.2          1.3         0.2  setosa                            1.5
4          4.6         3.1          1.5         0.2  setosa                            1.5
5          5.0         3.6          1.4         0.2  setosa                            1.4
6          5.4         3.9          1.7         0.4  setosa                            1.5
> all(created$Sepal.Length_less_Sepal.Width == (created$Sepal.Length - created$Sepal.Width))
[1] TRUE
> cat(code(created))
iris %>% dplyr::mutate(Sepal.Length_less_Sepal.Width = Sepal.Length -      Sepal.Width)
```

Missing to Categorical

Function: missingToCat()

Turn <NA>'s into "missing" and if applied to a numeric column, all non-missing numeric values will be turned into "observed"

```
> missing <- missingToCat(iris, vars = c("Species", "Sepal.Length"))
> head(missing)
  Sepal.Length Sepal.Length_miss Sepal.Width Petal.Length Petal.Width Species Species_miss
1          5.1          observed         3.5          1.4         0.2  setosa       setosa
2          4.9          observed         3.0          1.4         0.2  setosa       setosa
3          4.7          observed         3.2          1.3         0.2  setosa       setosa
4          4.6          observed         3.1          1.5         0.2  setosa       setosa
5          5.0          observed         3.6          1.4         0.2  setosa       setosa
6          5.4          observed         3.9          1.7         0.4  setosa       setosa
> cat(code(missing))
iris %>% tibble::add_column(Species_miss = forcats::fct_explicit_na(iris$Species,      na_level = "missing"), .after = "Sp
ecies") %>% tibble::add_column(Sepal.Length_miss = factor(ifelse(is.na(iris$Sepal.Length),      "missing", "observed")), .
after = "Sepal.Length")
```

Dataset Tab

Filter Dataset -> Levels of a categorical variable

Function: filterLevels()

Filter a dataset based off the values in a categorical column

```
> filtered <- filterLevels(iris, var = "Species", levels = c("versicolor", "virginica"))
> head(filtered)
  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          7.0         3.2          4.7         1.4 versicolor
2          6.4         3.2          4.5         1.5 versicolor
3          6.9         3.1          4.9         1.5 versicolor
4          5.5         2.3          4.0         1.3 versicolor
5          6.5         2.8          4.6         1.5 versicolor
6          5.7         2.8          4.5         1.3 versicolor
> unique(filtered$Species)
[1] versicolor virginica
Levels: versicolor virginica
> cat(code(filtered))
iris %>% dplyr::filter(Species %in% list("versicolor", "virginica")) %>%      dplyr::mutate(Species = factor(Species, leve
ls = list("versicolor",          "virginica")))
```

Filter Dataset -> Numeric condition

Function: filterNumeric()

Filter a dataset based off a logical condition in a numeric column

```
> filtered <- filterNumeric(iris, var = "Sepal.Length", op = "<=", num = 5)
> head(filtered)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          4.9         3.0          1.4         0.2  setosa
2          4.7         3.2          1.3         0.2  setosa
3          4.6         3.1          1.5         0.2  setosa
4          5.0         3.6          1.4         0.2  setosa
5          4.6         3.4          1.4         0.3  setosa
6          5.0         3.4          1.5         0.2  setosa
> all(filtered$Sepal.Length <= 5)
[1] TRUE
> cat(code(filtered))
iris %>% dplyr::filter(Sepal.Length <= 5)
```

Filter Dataset -> Row number

Function: filterRows()

Filter a dataset by slicing off specified row numbers

```
> filtered <- filterRows(iris, rows = c(1,4,5))
> head(filtered)
# A tibble: 6 x 5
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
         <dbl>       <dbl>        <dbl>       <dbl>  <fctr>
1          4.9         3.0          1.4         0.2  setosa
2          4.7         3.2          1.3         0.2  setosa
3          5.4         3.9          1.7         0.4  setosa
4          4.6         3.4          1.4         0.3  setosa
5          5.0         3.4          1.5         0.2  setosa
6          4.4         2.9          1.4         0.2  setosa
> cat(code(filtered))
iris %>% dplyr::slice(-c(1, 4, 5))
```

Filter Dataset -> Randomly

Function: filterRandom()

Sample X number of groups without replacement, each of size N

```
> filtered <- filterRandom(iris, n = 5, sample_size = 3)
> head(filtered)
  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species Sample.Number
1          7.7         2.6          6.9         2.3  virginica             1
2          6.0         2.2          4.0         1.0 versicolor             1
3          4.6         3.4          1.4         0.3     setosa             1
4          5.9         3.0          4.2         1.5 versicolor             2
5          5.0         3.4          1.5         0.2     setosa             2
6          6.4         2.7          5.3         1.9  virginica             2
> cat(code(filtered))
iris %>% dplyr::sample_n(3 * 5, replace = FALSE) %>% dplyr::mutate(Sample.Number = factor(rep(1:5,      each = 3)))
>
```

Sort data by variables

Function: sortVars()

Sort columns in either ascending or descending order. Sorts by up to 4 columns

```
> sorted <- sortVars(iris, vars = c("Sepal.Width", "Sepal.Length"), asc = c(TRUE, FALSE))
> head(sorted)
  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          5.0         2.0          3.5         1.0 versicolor
2          6.2         2.2          4.5         1.5 versicolor
3          6.0         2.2          4.0         1.0 versicolor
4          6.0         2.2          5.0         1.5  virginica
5          6.3         2.3          4.4         1.3 versicolor
6          5.5         2.3          4.0         1.3 versicolor
> cat(code(sorted))
iris %>% dplyr::arrange(Sepal.Width, desc(Sepal.Length))
```

Aggregate Data

Function: aggregateData()

Summarizes all the numeric variables by grouping them according to specified categorical variables and producing the summary of the mean, median, sum, sd and/or IQR.

Custom function made called countMissing() to give the number of missing values, functions identical to sum(is.na()) but has a is.na parameter which does nothing except for making the evaluation in aggregateData() work.

```
> aggregated <- aggregateData(iris, vars = c("Species"),
+                             summaries = c("mean", "sd", "iqr"))
> head(aggregated)
# A tibble: 3 x 18
     Species count Petal.Length.iqr Petal.Length.mean Petal.Length.sd Petal.Length.missing Petal.Width.iqr
      <fctr> <int>            <dbl>             <dbl>           <dbl>                <int>           <dbl>
1     setosa    50            0.175             1.462       0.1736640                    0             0.1
2 versicolor    50            0.600             4.260       0.4699110                    0             0.3
3  virginica    50            0.775             5.552       0.5518947                    0             0.5
# ... with 11 more variables: Petal.Width.mean <dbl>, Petal.Width.sd <dbl>, Petal.Width.missing <int>,
#   Sepal.Length.iqr <dbl>, Sepal.Length.mean <dbl>, Sepal.Length.sd <dbl>, Sepal.Length.missing <int>,
#   Sepal.Width.iqr <dbl>, Sepal.Width.mean <dbl>, Sepal.Width.sd <dbl>, Sepal.Width.missing <int>
> cat(code(aggregated))
iris %>% dplyr::group_by(Species) %>% dplyr::summarize(count = n(),      Petal.Length.iqr = IQR(Petal.Length, na.rm = TRUE), Petal.Length
.mean = mean(Petal.Length,          na.rm = TRUE), Petal.Length.sd = sd(Petal.Length, na.rm = TRUE),      Petal.Length.missing = countMis
sing(Petal.Length, na.rm = TRUE),      Petal.Width.iqr = IQR(Petal.Width, na.rm = TRUE), Petal.Width.mean = mean(Petal.Width,          na
.rm = TRUE), Petal.Width.sd = sd(Petal.Width, na.rm = TRUE),      Petal.Width.missing = countMissing(Petal.Width, na.rm = TRUE),      Sep
al.Length.iqr = IQR(Sepal.Length, na.rm = TRUE), Sepal.Length.mean = mean(Sepal.Length,          na.rm = TRUE), Sepal.Length.sd = sd(Sepa
l.Length, na.rm = TRUE),      Sepal.Length.missing = countMissing(Sepal.Length, na.rm = TRUE),      Sepal.Width.iqr = IQR(Sepal.Width, na
.rm = TRUE), Sepal.Width.mean = mean(Sepal.Width,          na.rm = TRUE), Sepal.Width.sd = sd(Sepal.Width, na.rm = TRUE),      Sepal.Widt
h.missing = countMissing(Sepal.Width, na.rm = TRUE))
```

Stack Variables

Function: stackVars()

Stack one or more columns converting the dataset to a tall format

```
> stacked <- stackVars(iris, vars = c("Species", "Sepal.Width"), key = "Variable", value = "Value")
Warning message:
attributes are not identical across measure variables;
they will be dropped
> head(stacked)
  Sepal.Length Petal.Length Petal.Width Variable  Value
1          5.1          1.4         0.2  Species setosa
2          4.9          1.4         0.2  Species setosa
3          4.7          1.3         0.2  Species setosa
4          4.6          1.5         0.2  Species setosa
5          5.0          1.4         0.2  Species setosa
6          5.4          1.7         0.4  Species setosa
> unique(stacked$Variable)
[1] "Species"     "Sepal.Width"
> cat(code(stacked))
iris %>% tidyr::gather(key = "Variable", value = "Value", Species,      Sepal.Width)
> |
```