RPandas Part 001

2023.03.22

Choosing a file

Choosing a file in Python with simple Dialog: easygui

```
# python code
#To install:
# pip install easygui
import easygui
#filename =easygui.fileopenbox()
#print(filename)
#easygui.egdemo()
```

Choosing a file in Python with simple Dialog: plyer

```
# python code
#To install:
# pip install plyer
#import plyer
#filename =plyer.filechooser.open_file()
#print(filename)
```

```
# R code
#filename =file.choose()
#print(filename)
```

read csv and xlsx files

csv

```
# python code
import pandas as pd
#import easygui
#filename = easygui.fileopenbox()
filename = "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris.csv"
df1=pd.read_csv(filename)
df1
```

```
##      Sepal_Length  Sepal_Width  Petal_Length  Petal_Width    Species
## 0             5.1          3.5           1.4          0.2     setosa
## 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
## ..            ...          ...           ...          ...        ...
## 145           6.7          3.0           5.2          2.3  virginica
## 146           6.3          2.5           5.0          1.9  virginica
## 147           6.5          3.0           5.2          2.0  virginica
## 148           6.2          3.4           5.4          2.3  virginica
## 149           5.9          3.0           5.1          1.8  virginica
##
## [150 rows x 5 columns]
```

```
# python code
#df1.dtypes
#df1.head()
#df1.tail(2)
#df1.columns
#df1.describe()
#df1["Petal_Length"]
#df1[0:2]
df1[df1.Petal_Length==1.4]
```

```
##     Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0             5.1          3.5           1.4          0.2  setosa
## 1             4.9          3.0           1.4          0.2  setosa
## 4             5.0          3.6           1.4          0.2  setosa
## 6             4.6          3.4           1.4          0.3  setosa
## 8             4.4          2.9           1.4          0.2  setosa
## 12            4.8          3.0           1.4          0.1  setosa
## 17            5.1          3.5           1.4          0.3  setosa
## 28            5.2          3.4           1.4          0.2  setosa
## 33            5.5          4.2           1.4          0.2  setosa
## 37            4.9          3.6           1.4          0.1  setosa
## 45            4.8          3.0           1.4          0.3  setosa
## 47            4.6          3.2           1.4          0.2  setosa
## 49            5.0          3.3           1.4          0.2  setosa
```

```
# R code
#filename = file.choose()
filename = "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris.csv"
df1=read.csv(filename)
head(df1)
```

```
##   Sepal_Length Sepal_Width Petal_Length Petal_Width Species
## 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
```

xlsx

```
#R code
library(openxlsx)
filename = "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris.xlsx"
df1=openxlsx::read.xlsx(filename)
head(df1)
```

R Code

```
##   Sepal_Length Sepal_Width Petal_Length Petal_Width Species
## 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
```

```
#R code
#read all sheets
library(openxlsx)
filename = "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris.xlsx"
SheetNames <- openxlsx::getSheetNames(filename)
SheetNames
```

```
## [1] "iris"   "Sheet1"
```

```
SheetList <- lapply(SheetNames,openxlsx::read.xlsx,xlsxFile=filename)
names(SheetList) <- SheetNames
SheetList$Sheet1[1:4,]
```

```
##   sheet2 Sepal_Width Petal_Length Petal_Width Species
## 1 sheet2         3.5          1.4         0.2  setosa
## 2 sheet2         3.0          1.4         0.2  setosa
## 3 sheet2         3.2          1.3         0.2  setosa
## 4 sheet2         3.1          1.5         0.2  setosa
```

```
SheetList$iris[1:4,]
```

```
##   Sepal_Length Sepal_Width Petal_Length Petal_Width Species
## 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
```

```
# write xlsx files
library(openxlsx)
wb <- createWorkbook()  #wb <- loadWorkbook("RawExcel.xlsx")
addWorksheet(wb, sheetName = "sheetname1")
```

```
writeData(wb, sheet = "sheetname1", x = SheetList$iris[1:4,])
addWorksheet(wb, sheetName = "sheetname2")
writeData(wb, sheet = "sheetname2", x = SheetList$Sheet1[1:4,])
#saveWorkbook(wb, "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris2.xlsx")
```

Python Code Python Code

```
import pandas as pd
xls = pd.ExcelFile('G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris2.xlsx')
xls.sheet_names
```

```
## ['sheetname1', 'sheetname2']
```

```
df1 = pd.read_excel(xls, xls.sheet_names[0])
df2 = pd.read_excel(xls, xls.sheet_names[1])
df1
```

```
##    Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 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
```

```
import pandas as pd
xls = pd.ExcelFile('G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris2.xlsx')
xls.sheet_names
```

```
## ['sheetname1', 'sheetname2']
```

```
df1 = pd.read_excel(xls, xls.sheet_names[0])
df2 = pd.read_excel(xls, xls.sheet_names[1])
df1
```

```
##    Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 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
```

```
dff=[pd.read_excel(xls, x) for x in xls.sheet_names]
```

```
import pandas as pd
dict_temp = pd.read_excel('G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris2.xlsx', sheet_name=
dict_temp['sheetname1']
```

```
##    Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 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
```

```
dict_temp['sheetname2']
```

```
##    sheet2  Sepal_Width  Petal_Length  Petal_Width Species
## 0  sheet2          3.5           1.4          0.2  setosa
## 1  sheet2          3.0           1.4          0.2  setosa
## 2  sheet2          3.2           1.3          0.2  setosa
## 3  sheet2          3.1           1.5          0.2  setosa
```

filter and select

filter

```
# python code
pl=1.4
qs="Petal_Length==@pl"
df1.query(qs)
```

```
##    Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 1           4.9          3.0           1.4          0.2  setosa
```

```
# python code
pw=.3
sp=["setosa","setosa1"]
qs="Species in @sp"\
 " and Petal_Width <= @pw"
df1.query(qs)
```

```
##    Sepal_Length  Sepal_Width  Petal_Length  Petal_Width Species
## 0           5.1          3.5           1.4          0.2  setosa
## 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
```

```
# R code
filename = "G:\\Python tutorial\\pythontutorial\\pythontutorial\\iris.csv"
df1=read.csv(filename)
pw=.3
sp=c("setosa","setosa1")
df1 %>%
  dplyr::filter(
    Species %in% sp
    ,Petal_Width <= pw
  ) %>% head()
```

```
##   Sepal_Length Sepal_Width Petal_Length Petal_Width Species
## 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          4.6         3.4          1.4         0.3  setosa
```

select

```
# python code
cl=["Sepal_Length","Petal_Width"]
df1[cl]
```

```
##    Sepal_Length  Petal_Width
## 0           5.1          0.2
## 1           4.9          0.2
## 2           4.7          0.2
## 3           4.6          0.2
```

```
# R code
cl=c("Sepal_Length","Petal_Width")
df1%>% dplyr::select(all_of(cl))  %>% head()
```

```
##   Sepal_Length Petal_Width
## 1          5.1         0.2
## 2          4.9         0.2
## 3          4.7         0.2
## 4          4.6         0.2
## 5          5.0         0.2
## 6          5.4         0.4
```

```
# df1%>% dplyr::select(Sepal_Length,Petal_Width)
```

starts_with

```
# R Code
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
iris %>%
   head(3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 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
```

```
iris %>%
  dplyr::select(starts_with('Sepal')) %>% head(3)
```

```
##   Sepal.Length Sepal.Width
## 1          5.1         3.5
## 2          4.9         3.0
## 3          4.7         3.2
```

ends_with

```
# R Code
iris %>%
  dplyr::select(ends_with('Length')) %>% head(3)
```

```
##   Sepal.Length Petal.Length
## 1          5.1          1.4
## 2          4.9          1.4
## 3          4.7          1.3
```

contains

```
# R Code
iris %>%
  dplyr::select(contains('Wid')) %>% head(3)
```

```
##   Sepal.Width Petal.Width
## 1         3.5         0.2
## 2         3.0         0.2
## 3         3.2         0.2
```

matches

```
# R Code
iris %>%
  dplyr::select(dplyr::matches("^(S)")) %>% head(3)
```

```
##   Sepal.Length Sepal.Width Species
## 1          5.1         3.5  setosa
## 2          4.9         3.0  setosa
## 3          4.7         3.2  setosa
```

```
iris %>%
  dplyr::select(dplyr::matches("^(Sepal.L|Sp)")) %>% head(3)
```

```
##   Sepal.Length Species
## 1          5.1  setosa
## 2          4.9  setosa
## 3          4.7  setosa
```

any_of

```
# R Code
iris %>%
  dplyr::select(dplyr::any_of(c('Sepal.Length','Sepal.Width','Petal.Length','Not_valid_name'))) %>% hea
```

```
##   Sepal.Length Sepal.Width Petal.Length
## 1          5.1         3.5          1.4
## 2          4.9         3.0          1.4
## 3          4.7         3.2          1.3
```

where

```
# R Code
iris %>%
  dplyr::select(dplyr::where(is.numeric)) %>% head(3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
```

siuba python

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# old approach: repeat name
cars[cars.cyl == 4]
# old approach: lambda
```

```
##     cyl   mpg   hp
## 2     4  22.8   93
## 7     4  24.4   62
## 8     4  22.8   95
## 17    4  32.4   66
## 18    4  30.4   52
## 19    4  33.9   65
## 20    4  21.5   97
## 25    4  27.3   66
## 26    4  26.0   91
## 27    4  30.4  113
## 31    4  21.4  109
```

```
cars[lambda _: _.cyl == 4]
# siu approach
```

```
##     cyl   mpg   hp
## 2     4  22.8   93
## 7     4  24.4   62
## 8     4  22.8   95
## 17    4  32.4   66
## 18    4  30.4   52
```

```
## 19    4  33.9   65
## 20    4  21.5   97
## 25    4  27.3   66
## 26    4  26.0   91
## 27    4  30.4  113
## 31    4  21.4  109
```

```
cars[_.cyl == 4]
```

```
##     cyl   mpg   hp
## 2     4  22.8   93
## 7     4  24.4   62
## 8     4  22.8   95
## 17    4  32.4   66
## 18    4  30.4   52
## 19    4  33.9   65
## 20    4  21.5   97
## 25    4  27.3   66
## 26    4  26.0   91
## 27    4  30.4  113
## 31    4  21.4  109
```

filter

siuba.org

```
import siuba
from siuba.data import cars
df2=(cars
    >> siuba.group_by(siuba._.cyl)
    >> siuba.filter(siuba._.mpg < siuba._.mpg.mean())
)
print(df2)
```

```
## (grouped data frame)
##     cyl   mpg   hp
## 2     4  22.8   93
## 5     6  18.1  105
## 6     8  14.3  245
## 7     4  24.4   62
## 8     4  22.8   95
## 9     6  19.2  123
## 10    6  17.8  123
## 14    8  10.4  205
## 15    8  10.4  215
## 16    8  14.7  230
## 20    4  21.5   97
## 23    8  13.3  245
## 26    4  26.0   91
## 29    6  19.7  175
## 30    8  15.0  335
## 31    4  21.4  109
```

```
import siuba
from siuba.data import cars
from siuba import _
df2=(cars
    >> siuba.group_by(_.cyl)
    >> siuba.filter(_.mpg < _.mpg.mean())
)
print(df2)
```

```
## (grouped data frame)
##     cyl   mpg   hp
## 2     4  22.8   93
## 5     6  18.1  105
## 6     8  14.3  245
## 7     4  24.4   62
## 8     4  22.8   95
## 9     6  19.2  123
## 10    6  17.8  123
## 14    8  10.4  205
## 15    8  10.4  215
## 16    8  14.7  230
## 20    4  21.5   97
## 23    8  13.3  245
## 26    4  26.0   91
## 29    6  19.7  175
## 30    8  15.0  335
## 31    4  21.4  109
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df2=(cars
    >> siuba.group_by( _.cyl )
    >> siuba.filter( _.mpg < _.mpg.mean() )
)
print(df2.head())
```

```
##     cyl   mpg   hp
## 2     4  22.8   93
## 5     6  18.1  105
## 6     8  14.3  245
## 7     4  24.4   62
## 8     4  22.8   95
## 9     6  19.2  123
## 10    6  17.8  123
## 14    8  10.4  205
## 15    8  10.4  215
## 16    8  14.7  230
## 20    4  21.5   97
## 23    8  13.3  245
## 26    4  26.0   91
## 29    6  19.7  175
```

isin (%in%)

```
import siuba
from siuba.data import cars
from siuba import _
import pandas as pd
df2=(cars
    >> siuba.filter( _.cyl.isin([8,6]))
)
print( df2)
```

```
##     cyl   mpg   hp
## 0     6  21.0  110
## 1     6  21.0  110
## 3     6  21.4  110
## 4     8  18.7  175
## 5     6  18.1  105
## 6     8  14.3  245
## 9     6  19.2  123
## 10    6  17.8  123
## 11    8  16.4  180
## 12    8  17.3  180
## 13    8  15.2  180
## 14    8  10.4  205
## 15    8  10.4  215
## 16    8  14.7  230
## 21    8  15.5  150
## 22    8  15.2  150
## 23    8  13.3  245
## 24    8  19.2  175
## 28    8  15.8  264
## 29    6  19.7  175
## 30    8  15.0  335
```

```
import siuba
from siuba.data import cars
from siuba import _
import pandas as pd
df2=(cars
    >> siuba.filter( _.cyl.isin([8,6]), _.hp.isin([123,180]))
)
print( df2)
```

```
##     cyl   mpg   hp
## 9     6  19.2  123
## 10    6  17.8  123
## 11    8  16.4  180
## 12    8  17.3  180
## 13    8  15.2  180
```

```
import siuba
from siuba.data import cars
from siuba import _
import pandas as pd
df3 = pd.DataFrame({
  'Country': ['Germany', 'France', 'Spain', 'Portugal']
  ,'num':[101,209,200,100]
  })
df4=(df3 >>
 siuba.filter(_.Country.isin(['Germany','Portugal']))
)
print(df4)
```

```
##     Country  num
## 0   Germany  101
## 3  Portugal  100
```

select

```
import siuba
from siuba.data import cars
from siuba import _
df2=(cars
    >> siuba.select(_.mpg , _.cyl )
)
print( df2.head() )
```

```
##     mpg  cyl
## 0  21.0    6
## 1  21.0    6
## 2  22.8    4
## 3  21.4    6
## 4  18.7    8
```

Select by name or position

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _

df2=(cars
    >> siuba.select(_.cyl, 1, ~_.hp )
)
print( df2.head() )
```

```
##    cyl   mpg
## 0    6  21.0
## 1    6  21.0
```

```
## 2    4  22.8
## 3    6  21.4
## 4    8  18.7
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _

df2=(cars
    >> siuba.select( -1,-2 )
)
print( df2.head() )
```

```
##     hp   mpg
## 0  110  21.0
## 1  110  21.0
## 2   93  22.8
## 3  110  21.4
## 4  175  18.7
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _

df2=(cars
    >> siuba.select( ~_.mpg )
)
print( df2.head() )
```

```
##    cyl   hp
## 0    6  110
## 1    6  110
## 2    4   93
## 3    6  110
## 4    8  175
```

Renaming columns

You can rename a specified column by using the equality operator (==). This operation takes the following form.

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
#_.new_name == _.old_name
df2=(cars
    >> siuba.select( _.hp2==_.hp, _.cyl2==_.cyl )
)
print( df2.head() )
```

```
##    hp2  cyl2
## 0  110     6
## 1  110     6
## 2   93     4
## 3  110     6
## 4  175     8
```

Use indexing (e.g. _["some_name"]) to refer to any column by name.

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_["start_name":"end_name"])
df2=(cars
    >> siuba.select(_['mpg' , 'hp'] )
)
print( df2.head() )
```

```
##     mpg   hp
## 0  21.0  110
## 1  21.0  110
## 2  22.8   93
## 3  21.4  110
## 4  18.7  175
```

Select by slice _[start:end]

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_["start_name":"end_name"])
df2=(cars
    >> siuba.select(_.cyl,_['mpg' : 'hp'] )
)
print( df2.head() )
```

```
##    cyl   mpg   hp
## 0    6  21.0  110
## 1    6  21.0  110
## 2    4  22.8   93
## 3    6  21.4  110
## 4    8  18.7  175
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
```

```
# select(_["start_name":"end_name"])
df2=(cars
    >> siuba.select(_.cyl,_[1:3])
)
print( df2.head() )
```

```
##    cyl   mpg   hp
## 0    6  21.0  110
## 1    6  21.0  110
## 2    4  22.8   93
## 3    6  21.4  110
## 4    8  18.7  175
```

Exclusion

You can exclude slice selections using the ~ operator.

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(~_["start_name":"end_name"])
df2=(cars
    >> siuba.select(_.cyl,~_['mpg' : 'hp'] )
)
print( df2.head() )
```

```
##    cyl
## 0    6
## 1    6
## 2    4
## 3    6
## 4    8
```

Select by pattern (e.g. endswith)

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.endswith("mm"))
df2=(cars
    >> siuba.select(_.cyl,_.endswith('p') )
)
print( df2.head() )
```

```
##    cyl   hp
## 0    6  110
## 1    6  110
```

```
## 2    4   93
## 3    6  110
## 4    8  175
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.endswith("mm"))
df2=(cars
    >> siuba.select(~_.endswith('p') )
)
print( df2.head() )
```

```
##    cyl   mpg
## 0    6  21.0
## 1    6  21.0
## 2    4  22.8
## 3    6  21.4
## 4    8  18.7
```

Select by pattern (e.g. startswith)

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.startswith("mm"))
df2=(cars
    >> siuba.select(_.cyl,_.startswith('m') )
)
print( df2.head() )
```

```
##    cyl   mpg
## 0    6  21.0
## 1    6  21.0
## 2    4  22.8
## 3    6  21.4
## 4    8  18.7
```

Select by pattern (e.g. contains)

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(_.cyl,_.contains('p') )
)
print( df2.head() )
```

```
##    cyl   mpg   hp
## 0    6  21.0  110
## 1    6  21.0  110
## 2    4  22.8   93
## 3    6  21.4  110
## 4    8  18.7  175
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(_.contains('h|c') )
)
print( df2.head() )
```

```
##    cyl   hp
## 0    6  110
## 1    6  110
## 2    4   93
## 3    6  110
## 4    8  175
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(~_.contains('h|c') )
)
print( df2.head() )
```

```
##     mpg
## 0  21.0
## 1  21.0
## 2  22.8
## 3  21.4
## 4  18.7
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(_.contains('^c|g$') )
)
print( df2.head() )
```

```
##    cyl   mpg
```

```
## 0    6  21.0
## 1    6  21.0
## 2    4  22.8
## 3    6  21.4
## 4    8  18.7
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(_.contains('.{2}g$') )
)
print( df2.head() )
```

```
##     mpg
## 0  21.0
## 1  21.0
## 2  22.8
## 3  21.4
## 4  18.7
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.select(_.contains('p.+') )
)
print( df2.head() )
```

```
##     mpg
## 0  21.0
## 1  21.0
## 2  22.8
## 3  21.4
## 4  18.7
```

mutate

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.mutate(mpg_std=(_.mpg-_.mpg.mean())/_.mpg.std(),mpg_hp=_.mpg*_.hp,hp_per_cyl = _.hp / _.cyl
)
print( df2.head() )
```

```
##    cyl   mpg   hp   mpg_std  mpg_hp  hp_per_cyl
## 0    6  21.0  110  0.150885  2310.0   18.333333
## 1    6  21.0  110  0.150885  2310.0   18.333333
## 2    4  22.8   93  0.449543  2120.4   23.250000
## 3    6  21.4  110  0.217253  2354.0   18.333333
## 4    8  18.7  175 -0.230735  3272.5   21.875000
```

Grouped mutates

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df5=(cars
  >> siuba.group_by( _.cyl )
  >> siuba.mutate(
       hp_mean = _.hp.mean(),
       demeaned_hp = _.hp - _.hp_mean
     )
  )
print(df5)
```

```
## (grouped data frame)
##     cyl   mpg   hp     hp_mean  demeaned_hp
## 0     6  21.0  110  122.285714   -12.285714
## 1     6  21.0  110  122.285714   -12.285714
## 2     4  22.8   93   82.636364    10.363636
## 3     6  21.4  110  122.285714   -12.285714
## 4     8  18.7  175  209.214286   -34.214286
## 5     6  18.1  105  122.285714   -17.285714
## 6     8  14.3  245  209.214286    35.785714
## 7     4  24.4   62   82.636364   -20.636364
## 8     4  22.8   95   82.636364    12.363636
## 9     6  19.2  123  122.285714     0.714286
## 10    6  17.8  123  122.285714     0.714286
## 11    8  16.4  180  209.214286   -29.214286
## 12    8  17.3  180  209.214286   -29.214286
## 13    8  15.2  180  209.214286   -29.214286
## 14    8  10.4  205  209.214286    -4.214286
## 15    8  10.4  215  209.214286     5.785714
## 16    8  14.7  230  209.214286    20.785714
## 17    4  32.4   66   82.636364   -16.636364
## 18    4  30.4   52   82.636364   -30.636364
## 19    4  33.9   65   82.636364   -17.636364
## 20    4  21.5   97   82.636364    14.363636
## 21    8  15.5  150  209.214286   -59.214286
## 22    8  15.2  150  209.214286   -59.214286
## 23    8  13.3  245  209.214286    35.785714
## 24    8  19.2  175  209.214286   -34.214286
## 25    4  27.3   66   82.636364   -16.636364
## 26    4  26.0   91   82.636364     8.363636
## 27    4  30.4  113   82.636364    30.363636
```

```
## 28    8  15.8  264  209.214286    54.785714
## 29    6  19.7  175  122.285714    52.714286
## 30    8  15.0  335  209.214286   125.785714
## 31    4  21.4  109   82.636364    26.363636
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','B','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=(df6
  >> siuba.group_by( _.g1 )
  >> siuba.mutate(
       vl_mean = _.vl.mean(),
       vl_std = _.vl.std()
       ,vl_z=(_.vl-_.vl.mean())/_.vl.std()
     )
  )
print(df5)
```

```
## (grouped data frame)
##   g1 g2  vl  vl_mean  vl_std  vl_z
## 0  a  A   1      2.0     1.0  -1.0
## 1  a  A   2      2.0     1.0   0.0
## 2  a  B   3      2.0     1.0   1.0
## 3  b  A   7      8.0     1.0  -1.0
## 4  b  B   8      8.0     1.0   0.0
## 5  b  B   9      8.0     1.0   1.0
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','B','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=(df6
  >> siuba.group_by( _.g1 ,_.g2)
  >> siuba.mutate(
       vl_mean = _.vl.mean(),
       vl_std = _.vl.std()
       ,vl_z=(_.vl-_.vl.mean())/_.vl.std()
     )
```

```
  )
print(df5)
```

```
## (grouped data frame)
##   g1 g2  vl  vl_mean    vl_std      vl_z
## 0  a  A   1      1.5  0.707107 -0.707107
## 1  a  A   2      1.5  0.707107  0.707107
## 2  a  B   3      3.0       NaN       NaN
## 3  b  A   7      7.0       NaN       NaN
## 4  b  B   8      8.5  0.707107 -0.707107
## 5  b  B   9      8.5  0.707107  0.707107
```

shift

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','B','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=df5=(df6
  >> siuba.mutate(
       vl_lag = _.vl - _.vl.shift(1)
     )
  )
print(df5)
```

```
##   g1 g2  vl  vl_lag
## 0  a  A   1     NaN
## 1  a  A   2     1.0
## 2  a  B   3     1.0
## 3  b  A   7     4.0
## 4  b  B   8     1.0
## 5  b  B   9     1.0
```

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','B','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
```

```
df5=df5=(df6
  >> siuba.mutate(
       vl_lag = _.vl - _.vl.shift(-1)
     )
  )
print(df5)
```

```
##   g1 g2  vl  vl_lag
## 0  a  A   1    -1.0
## 1  a  A   2    -1.0
## 2  a  B   3    -4.0
## 3  b  A   7    -1.0
## 4  b  B   8    -1.0
## 5  b  B   9     NaN
```

Grouped shifts

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=df5=(df6
  >> siuba.group_by(_.g1)
  >> siuba.mutate(
       vl_lag = _.vl - _.vl.shift(1)
     )
  )
print(df5)
```

```
## (grouped data frame)
##   g1  vl  vl_lag
## 0  a   1     NaN
## 1  a   2     1.0
## 2  a   3     1.0
## 3  b   7     NaN
## 4  b   8     1.0
## 5  b   9     1.0
```

summarize

Summarize over all rows

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
# select(_.species, _.contains("mm"))
df2=(cars
    >> siuba.summarize(mpg_sum=_.mpg.sum(),mpg_mean=_.mpg.mean(),mpg_min=_.mpg.min(),mpg_max=_.mpg.max()
     , mpg_std=_.mpg.std(), mpg_median=_.mpg.median(),mpg_count=_.mpg.count())
)
print( df2.head() )
```

```
##    mpg_sum   mpg_mean  mpg_min  mpg_max   mpg_std  mpg_median  mpg_count
## 0    642.9  20.090625     10.4     33.9  6.026948        19.2         32
```

Summarize over groups

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','B','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=(df6
  >> siuba.group_by( _.g1 )
  >> siuba.summarize(
       vl_mean = _.vl.mean()
       ,vl_std=_.vl.std()
       ,vl_min=_.vl.min()
       ,vl_max=_.vl.max()
       ,vl_median=_.vl.median()
     )
  )
print(df5)
```

```
##   g1  vl_mean  vl_std  vl_min  vl_max  vl_median
## 0  a      2.0     1.0       1       3        2.0
## 1  b      8.0     1.0       7       9        8.0
```

```
import pandas as pd
import siuba
from siuba.data import mtcars
from siuba import _
df5=(mtcars
  >> siuba.group_by(_.cyl)
  >> siuba.summarize(
       avg = _.mpg.mean(),
```

```
      range = _.mpg.max() - _.mpg.min(),
      avg_per_cyl = (_.mpg / _.cyl).mean()
  )
)
print(df5)
```

```
##    cyl        avg  range  avg_per_cyl
## 0    4  26.663636   12.5     6.665909
## 1    6  19.742857    3.6     3.290476
## 2    8  15.100000    8.8     1.887500
```

Group by an expression

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
   'g1':['a','a','a','b','b','b']
    ,'vl':[1,2,3,7,8,9]
 }
 )
df5=(df6
  >> siuba.group_by( g1_temp= _.vl>3 )
  >> siuba.summarize(
       vl_mean = _.vl.mean()
       ,vl_std=_.vl.std()
       ,vl_min=_.vl.min()
       ,vl_max=_.vl.max()
       ,vl_median=_.vl.median()
     )
  )
print(df5)
```

```
##    g1_temp  vl_mean  vl_std  vl_min  vl_max  vl_median
## 0    False      2.0     1.0       1       3        2.0
## 1     True      8.0     1.0       7       9        8.0
```

group_by(high_hp = _.hp > 300)

Count rows

```
import pandas as pd
import siuba
from siuba.data import cars
from siuba import _
df6=pd.DataFrame(
 {
```

```
   'g1':['a','a','a','b','b','b']
   ,'g2':['A','A','B','A','A','B']
   ,'vl':[1,2,3,7,8,9]
 }
 )
df5=(df6
  >> siuba.group_by( _.g1 )
  >> siuba.summarize(
      count=_.shape[0]
     )
  )
print(df5)
```

```
##   g1  count
## 0  a      3
## 1  b      3
```

```
df5=(df6
  >> siuba.group_by( _.g2 )
  >> siuba.summarize(
      count=_.shape[0]
     )
  )
print(df5)
```

```
##   g2  count
## 0  A      4
## 1  B      2
```

sort_values

```
from siuba import _, count
from siuba.data import mtcars

df3=(mtcars
    >> count(_.cyl)          # this is a siuba verb
    >> _.sort_values("n")    # this is a pandas method
)
print(df3)
```

```
##    cyl   n
## 1    6   7
## 0    4  11
## 2    8  14
```

shape

```
from siuba import _, count
from siuba.data import mtcars

df3=(mtcars
    >> _.shape
)
print(df3)
```

```
## (32, 11)
```

```
df3=(mtcars
    >> _.shape[0]
)
print(df3)
```

```
## 32
```

```
df3=(mtcars
    >> _.shape[1]
)
print(df3)
```

```
## 11
```

Google colab overrides __ (import __ as X)

```
from siuba import _ as XX, filter
from siuba.data import mtcars

df3=mtcars >> filter(XX.mpg > 30)
print(df3)
```

```
##      mpg  cyl  disp   hp  drat     wt   qsec  vs  am  gear  carb
## 17  32.4    4  78.7   66  4.08  2.200  19.47   1   1     4     1
## 18  30.4    4  75.7   52  4.93  1.615  18.52   1   1     4     2
## 19  33.9    4  71.1   65  4.22  1.835  19.90   1   1     4     1
## 27  30.4    4  95.1  113  3.77  1.513  16.90   1   1     5     2
```

Dropping NA values

Use pandas.isna() / pandas.notna() to determine whether a value is considered to be NA.

```
df4 = pd.DataFrame({
    "x": [True, False, None,True, False, None],
    "value":[0,5,6,3,9,8]
    })

print(df4)
```

```
##        x  value
## 0   True      0
## 1  False      5
## 2   None      6
## 3   True      3
## 4  False      9
## 5   None      8
```

```
print(pd.isna(df4.x))
```

```
## 0    False
## 1    False
## 2     True
## 3    False
## 4    False
## 5     True
## Name: x, dtype: bool
```

```
print(df4[pd.isna(df4.x)])
```

```
##       x  value
## 2  None      6
## 5  None      8
```

```
print(df4[pd.notna(df4.x)])
```

```
##        x  value
## 0   True      0
## 1  False      5
## 3   True      3
## 4  False      9
```

```
import pandas as pd
from siuba import _, count
from siuba.data import mtcars

df5=df4 >> siuba.filter(_.x)
print(df5)
```

```
##       x  value
## 0  True      0
## 3  True      3
```

```
import pandas as pd
from siuba import _, count
from siuba.data import mtcars

df5=df4 >> siuba.filter(_.x.notna())
print(df5)
```

```
##        x  value
## 0   True      0
## 1  False      5
## 3   True      3
## 4  False      9
```

```
import pandas as pd
from siuba import _, count
from siuba.data import mtcars

df5=df4 >> siuba.filter(_.x.notna(),_.value)
print(df5)
```

```
##        x  value
## 1  False      5
## 3   True      3
## 4  False      9
```

```
import pandas as pd
from siuba import _, count
from siuba.data import mtcars

df5=df4 >> siuba.filter(_.x,_.value)
print(df5)
```

```
##       x  value
## 3  True      3
```

arrange

```
import pandas as pd
import siuba
from siuba import _, count
from siuba.data import cars

df5= cars >> siuba.arrange(-_.hp)
print(df5.head())
```

```
##     cyl   mpg   hp
## 30    8  15.0  335
## 28    8  15.8  264
## 6     8  14.3  245
## 23    8  13.3  245
## 16    8  14.7  230
```

```
import pandas as pd
import siuba
from siuba import _, count
from siuba.data import cars

df5= cars >> siuba.arrange(_.cyl,-_.mpg)
print(df5)
```

```
##     cyl   mpg   hp
## 19    4  33.9   65
## 17    4  32.4   66
## 18    4  30.4   52
## 27    4  30.4  113
## 25    4  27.3   66
## 26    4  26.0   91
## 7     4  24.4   62
## 2     4  22.8   93
## 8     4  22.8   95
## 20    4  21.5   97
## 31    4  21.4  109
## 3     6  21.4  110
## 0     6  21.0  110
## 1     6  21.0  110
## 29    6  19.7  175
## 9     6  19.2  123
## 5     6  18.1  105
## 10    6  17.8  123
## 24    8  19.2  175
## 4     8  18.7  175
## 12    8  17.3  180
## 11    8  16.4  180
## 28    8  15.8  264
## 21    8  15.5  150
## 13    8  15.2  180
## 22    8  15.2  150
## 30    8  15.0  335
## 16    8  14.7  230
## 6     8  14.3  245
## 23    8  13.3  245
## 14    8  10.4  205
## 15    8  10.4  215
```

```
df5=cars >> siuba.arrange(_.hp / _.cyl)
print(df5)
```

```
##     cyl   mpg   hp
## 18    4  30.4   52
## 7     4  24.4   62
## 19    4  33.9   65
## 17    4  32.4   66
## 25    4  27.3   66
## 5     6  18.1  105
## 0     6  21.0  110
## 1     6  21.0  110
## 3     6  21.4  110
## 21    8  15.5  150
## 22    8  15.2  150
## 9     6  19.2  123
## 10    6  17.8  123
## 4     8  18.7  175
## 24    8  19.2  175
## 11    8  16.4  180
```

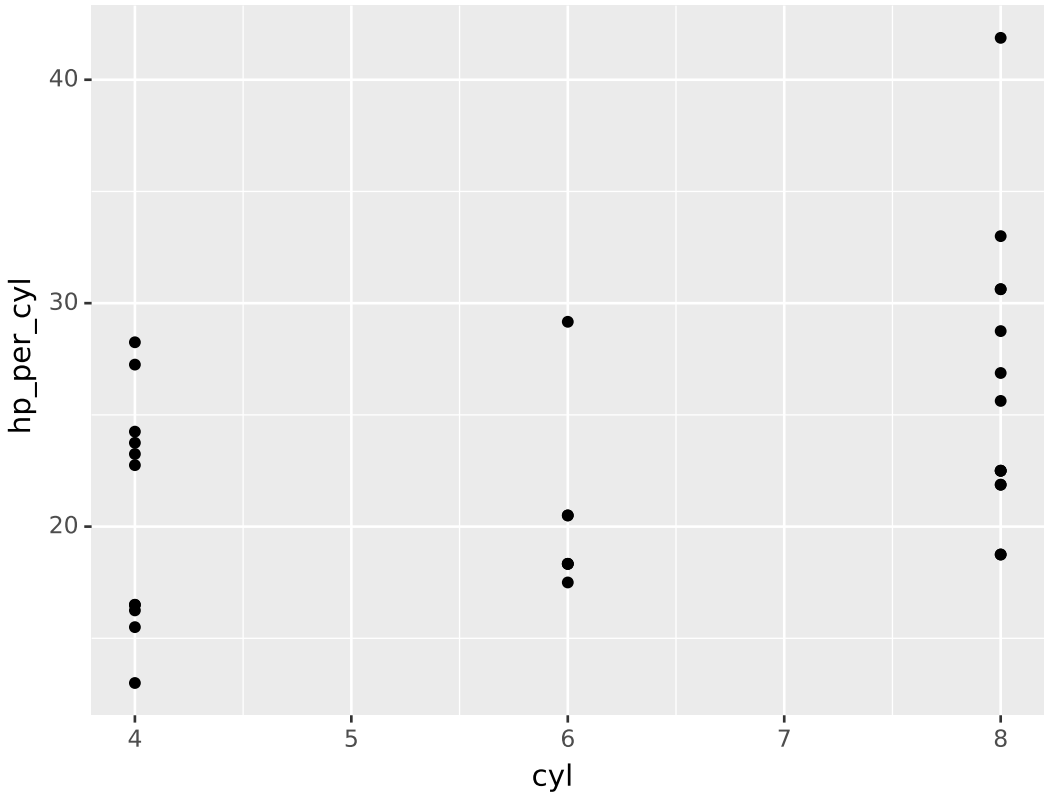
```
## 12    8  17.3  180
## 13    8  15.2  180
## 26    4  26.0   91
## 2     4  22.8   93
## 8     4  22.8   95
## 20    4  21.5   97
## 14    8  10.4  205
## 15    8  10.4  215
## 31    4  21.4  109
## 27    4  30.4  113
## 16    8  14.7  230
## 29    6  19.7  175
## 6     8  14.3  245
## 23    8  13.3  245
## 28    8  15.8  264
## 30    8  15.0  335
```

Using with plotnine

```
from siuba import mutate, _
from siuba.data import mtcars
from plotnine import ggplot, aes, geom_point

(mtcars
  >> mutate(hp_per_cyl = _.hp / _.cyl)
  >> ggplot(aes("cyl", "hp_per_cyl"))
   + geom_point()
)
```

```
## <ggplot: (113010863156)>
```



Call external functions

```
import pandas as pd
from siuba import _, mutate
from siuba.siu import call

my_dates = pd.DataFrame({"date": ["2021-01-01", "2021-01-02"]})

pd.to_datetime(my_dates.date)
```

```
## 0   2021-01-01
## 1   2021-01-02
## Name: date, dtype: datetime64[ns]
```

```
my_dates >> mutate(parsed = _.date) >> _.parsed
```

```
## 0    2021-01-01
## 1    2021-01-02
## Name: parsed, dtype: object
```

```
my_dates >> mutate(parsed = call(pd.to_datetime, _.date))
```

```
##          date     parsed
## 0  2021-01-01 2021-01-01
## 1  2021-01-02 2021-01-02
```

sql basic

https://siuba.org/guide/basics-sql.html

pivot/melt

melt - pivot_longer

```
# python code
import pandas as pd
df2 = pd.DataFrame({'A': {0: 'a', 1: 'b', 2: 'c'},
                    'B': {0: 1, 1: 3, 2: 5},
                    'C': {0: 2, 1: 4, 2: 6}})
df2.melt(id_vars='A')
```

```
##    A variable  value
## 0  a        B      1
## 1  b        B      3
## 2  c        B      5
## 3  a        C      2
## 4  b        C      4
## 5  c        C      6
```

```
df2.melt(id_vars='A', value_vars=['B','C'], var_name='BC', value_name='value')
```

```
##    A BC  value
## 0  a  B      1
## 1  b  B      3
## 2  c  B      5
## 3  a  C      2
## 4  b  C      4
## 5  c  C      6
```

```
# R code
library(tidyr)
df2 = data.frame(
  A=c('a','b','c')
  ,B=c(1,3,5)
  ,C=c(2,4,6)
  )
df2 %>%
  pivot_longer(B:C,names_to = 'BC',values_to = 'value')  %>% head()
```

```
## # A tibble: 6 x 3
##   A     BC    value
##   <chr> <chr> <dbl>
## 1 a     B         1
## 2 a     C         2
## 3 b     B         3
## 4 b     C         4
## 5 c     B         5
## 6 c     C         6
```

```r
# R code
library(tidyr)
df2 = data.frame(
  A=c('a', 'a','b', 'b', 'c','c')
  ,B=c('A', 'B','A', 'B', 'A','B')
  ,D=c( 1, 3, 5,7,9,11)
  ,E=c(2, 4, 6,8,10,12)
  )
df2 %>%
  tidyr::pivot_longer(cols = any_of(c('D','E')),names_to = "DE",values_to  = "value")  %>% head()
```

```
## # A tibble: 6 x 4
##   A     B     DE    value
##   <chr> <chr> <chr> <dbl>
## 1 a     A     D         1
## 2 a     A     E         2
## 3 a     B     D         3
## 4 a     B     E         4
## 5 b     A     D         5
## 6 b     A     E         6
```

pivot_wider

```python
# python code
import pandas as pd
df2 = pd.DataFrame({'A': {0: 'a', 1: 'b', 2: 'c'},
                    'B': {0: 1, 1: 3, 2: 5},
                    'C': {0: 2, 1: 4, 2: 6}})
#print(df2)
df2_melt=df2.melt(id_vars='A', value_vars=['B','C'], var_name='BC', value_name='value')
#print(df2_melt)
df_pivot=df2_melt.pivot(index='A', columns=['BC'])#, values='value')
df2_r = df_pivot.reset_index(None)
df2_r.columns = ['A', 'B', 'C']
print(df2_r)
```

```
##    A  B  C
## 0  a  1  2
## 1  b  3  4
## 2  c  5  6
```

```
df2_r.columns=df2.columns.values
print(df2.columns.values)
```

```
## ['A' 'B' 'C']
```

```
print(df2_r)
```

```
##    A  B  C
## 0  a  1  2
## 1  b  3  4
## 2  c  5  6
```

```
# R code
library(tidyr)
df2 = data.frame(
  A=c('a','b','c')
  ,B=c(1,3,5)
  ,C=c(2,4,6)
)
df2_melt<-df2 %>%
  tidyr::pivot_longer(cols = any_of(c('B','C')),names_to = "BC",values_to  = "value")
df_pivot <- df2_melt %>%
  tidyr::pivot_wider(id_cols = A, names_from = BC,values_from = value )
df_pivot  %>% head()
```

```
## # A tibble: 3 x 3
##   A         B     C
##   <chr> <dbl> <dbl>
## 1 a         1     2
## 2 b         3     4
## 3 c         5     6
```

The across function

across

- across() makes it easy to apply the same transformation to multiple columns, allowing you to use select() semantics inside in "data-masking" functions like summarise() and mutate().
- if_any() and if_all() are used to apply the same predicate function to a selection of columns and combine the results into a single logical vector.
- across() supersedes the family of dplyr "scoped variants" like summarise_at(), summarise_if(), and summarise_all() and therefore these functions will not be implemented in poorman. across: Apply a function (or functions) across multiple columns

Usage

- across(.cols = everything(), .fns = NULL, . . . , .names = NULL)
- if_any(.cols, .fns = NULL, . . . , .names = NULL)
- if_all(.cols, .fns = NULL, . . . , .names = NULL)

Arguments

.fns Functions to apply to each of the selected columns. Possible values are:

- NULL, to returns the columns untransformed.
- A function, e.g. mean.
- A lambda, e.g. ~ mean(.x, na.rm = TRUE)
- A list of functions/lambdas, e.g. list(mean = mean, n_miss = ~ sum(is.na(.x))

Within these functions you can use cur_column() and cur_group() to access the current column and grouping keys respectively.

... Additional arguments for the function calls in .fns.

.names character(n). Currently limited to specifying a vector of names to use for the outputs.

cols, .cols Columns to transform. Because across() is used within functions like summarise() and mutate(), you can't select or compute upon grouping variables.

Value

- across() returns a data.frame with one column for each column in .cols and each function in .fns.
- if_any() and if_all() return a logical vector.

How to use across

There are four columns and I want to quickly get the mean of these columns for each category. First, here's how I might do this without across:

```
# R code
iris  %>%
  group_by(Species) %>%
  summarise(
    Sepal.Length = mean(Sepal.Length, na.rm = TRUE),
    Sepal.Width = mean(Sepal.Width, na.rm = TRUE),
    Petal.Width = mean(Petal.Width, na.rm = TRUE),
    Petal.Length = mean(Petal.Length, na.rm = TRUE)
    )  %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Width Petal.Length
##   <fct>             <dbl>       <dbl>       <dbl>        <dbl>
## 1 setosa             5.01        3.43       0.246         1.46
## 2 versicolor         5.94        2.77       1.33          4.26
## 3 virginica          6.59        2.97       2.03          5.55
```

Which works fine. But imagine if instead of four columns there were 10 or 20 or 100! It would quickly get tedious to add a new line for each column. Here's where across comes in:

```r
# R code
iris  %>%
  group_by(Species) %>%

  summarise(across(c(Sepal.Length, Sepal.Width, Petal.Length,Petal.Width),mean,na.rm = TRUE)) %>% head()
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(...)`.
## i In group 1: `Species = setosa`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Length Petal.Width
##   <fct>             <dbl>       <dbl>        <dbl>       <dbl>
## 1 setosa             5.01        3.43         1.46       0.246
## 2 versicolor         5.94        2.77         4.26       1.33
## 3 virginica          6.59        2.97         5.55       2.03
```

Much more efficient. We give across a vector of column names followed by the function (in this case mean) followed by any other arguments we want to apply to the function.

:

: for selecting a range of consecutive variables.

```r
# R code
iris  %>%
  group_by(Species) %>%
       summarise(across(c(Sepal.Length:Petal.Width), mean, na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Length Petal.Width
##   <fct>             <dbl>       <dbl>        <dbl>       <dbl>
## 1 setosa             5.01        3.43         1.46       0.246
## 2 versicolor         5.94        2.77         4.26       1.33
## 3 virginica          6.59        2.97         5.55       2.03
```

!

! for taking the complement of a set of variables.

```r
# R code
iris  %>%
  group_by(Species) %>%
       summarise(across(!c(Petal.Width), mean, na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 4
##   Species    Sepal.Length Sepal.Width Petal.Length
##   <fct>             <dbl>       <dbl>        <dbl>
## 1 setosa             5.01        3.43         1.46
## 2 versicolor         5.94        2.77         4.26
## 3 virginica          6.59        2.97         5.55
```

& and |

& and |for selecting the intersection or the union of two sets of variables.

```r
# R code
iris  %>%
  group_by(Species) %>%
       summarise(across(ends_with('Length') & !c(Petal.Length, Petal.Width), mean, na.rm = TRUE))  %>% h
```

```
## # A tibble: 3 x 2
##   Species    Sepal.Length
##   <fct>             <dbl>
## 1 setosa             5.01
## 2 versicolor         5.94
## 3 virginica          6.59
```

c()

c() for combining selections.

```r
# R code
iris  %>%
  group_by(Species) %>%

  summarise(across(c(Sepal.Length, Sepal.Width, Petal.Length,Petal.Width),mean,na.rm = TRUE)) %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Length Petal.Width
##   <fct>             <dbl>       <dbl>        <dbl>       <dbl>
## 1 setosa             5.01        3.43         1.46       0.246
## 2 versicolor         5.94        2.77         4.26       1.33
## 3 virginica          6.59        2.97         5.55       2.03
```

starts_with()

starts_with(): Starts with a prefix.

```r
# R code
iris  %>%
  group_by(Species) %>%

  summarise(across(starts_with("S"),mean,na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 3
##   Species    Sepal.Length Sepal.Width
##   <fct>             <dbl>       <dbl>
## 1 setosa             5.01        3.43
## 2 versicolor         5.94        2.77
## 3 virginica          6.59        2.97
```

ends_with()

ends_with(): Ends with a suffix.

```r
# R code
iris  %>%
  group_by(Species) %>%

  summarise(across(ends_with("dth"),mean,na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 3
##   Species    Sepal.Width Petal.Width
##   <fct>            <dbl>       <dbl>
## 1 setosa            3.43       0.246
## 2 versicolor        2.77       1.33
## 3 virginica         2.97       2.03
```

contains()

contains(): Contains a literal string.

```r
# R code
iris  %>%
  group_by(Species) %>%
    summarise(across(contains('Length'),mean,na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 3
##   Species    Sepal.Length Petal.Length
##   <fct>             <dbl>        <dbl>
## 1 setosa             5.01         1.46
## 2 versicolor         5.94         4.26
## 3 virginica          6.59         5.55
```

matches()

matches(): Matches a regular expression.

```r
# R code
iris  %>%
  group_by(Species) %>%
    summarise(across(matches('^(S|P)'),mean,na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Length Petal.Width
##   <fct>             <dbl>       <dbl>        <dbl>       <dbl>
## 1 setosa             5.01        3.43         1.46       0.246
## 2 versicolor         5.94        2.77         4.26       1.33
## 3 virginica          6.59        2.97         5.55       2.03
```

num_range()

num_range(): Matches a numerical range like x01, x02, x03.

```r
# R code
df <- as.data.frame(matrix(1:24, nrow = 3))
df  %>% head()
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8
## 1  1  4  7 10 13 16 19 22
## 2  2  5  8 11 14 17 20 23
## 3  3  6  9 12 15 18 21 24
```

```r
df %>% select(num_range("V", seq(1, 1000, by = 3)))  %>% head()
```

```
##   V1 V4 V7
## 1  1 10 19
## 2  2 11 20
## 3  3 12 21
```

```r
# R code
df <- data.frame(id=c("a","a","b"), tot_1=4:6, tot_2=8:10, tot_3=11:13, tot_4=33:35,tot_5=22:24)
df  %>% head()
```

```
##   id tot_1 tot_2 tot_3 tot_4 tot_5
## 1  a     4     8    11    33    22
## 2  a     5     9    12    34    23
## 3  b     6    10    13    35    24
```

```r
df %>% group_by(id) %>%
  mutate(across(.cols = num_range("tot_", seq(1, 5, by = 2)),mean,na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 6
## # Groups:   id [2]
##   id    tot_1 tot_2 tot_3 tot_4 tot_5
##   <chr> <dbl> <int> <dbl> <int> <dbl>
## 1 a       4.5     8  11.5    33  22.5
## 2 a       4.5     9  11.5    34  22.5
## 3 b       6      10  13      35  24
```

```r
# R code
df %>% group_by(id) %>%
  summarise(across(.cols = num_range(prefix="tot_", range=seq(1, 5, by = 2)),mean,na.rm = TRUE))  %>% he
```

```
## # A tibble: 2 x 4
##   id    tot_1 tot_3 tot_5
##   <chr> <dbl> <dbl> <dbl>
## 1 a       4.5  11.5  22.5
## 2 b       6    13    24
```

all_of()

all_of(): Matches variable names in a character vector. All names must be present, otherwise an out-of-bounds error is thrown.

```r
# R code
iris  %>%
  group_by(Species) %>%
    summarise(across(all_of(c('Sepal.Length','Sepal.Width','Petal.Length')),mean,na.rm = TRUE))  %>% hea
```

```
## # A tibble: 3 x 4
##   Species    Sepal.Length Sepal.Width Petal.Length
##   <fct>             <dbl>       <dbl>        <dbl>
## 1 setosa             5.01        3.43         1.46
## 2 versicolor         5.94        2.77         4.26
## 3 virginica          6.59        2.97         5.55
```

any_of()

any_of(): Same as all_of(), except that no error is thrown for names that don't exist.

```r
# R code
iris  %>%
  group_by(Species) %>%
    summarise(across(any_of(c('Sepal.Length','Sepal.Width','Petal.Length','Not_valid_name')),mean,na.rm
```

```
## # A tibble: 3 x 4
##   Species    Sepal.Length Sepal.Width Petal.Length
##   <fct>             <dbl>       <dbl>        <dbl>
## 1 setosa             5.01        3.43         1.46
## 2 versicolor         5.94        2.77         4.26
## 3 virginica          6.59        2.97         5.55
```

where()

where(): Applies a function to all variables and selects those for which the function returns TRUE.

```r
# R code
iris  %>%
  group_by(Species) %>%
       summarise(across(where(is.numeric), mean, na.rm = TRUE))  %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length Sepal.Width Petal.Length Petal.Width
##   <fct>             <dbl>       <dbl>        <dbl>       <dbl>
## 1 setosa             5.01        3.43         1.46       0.246
## 2 versicolor         5.94        2.77         4.26       1.33
## 3 virginica          6.59        2.97         5.55       2.03
```

Using in-line functions with across

Let's look at an example of summarizing the columns using a custom function (rather than n_distinct()). I usually do this using the tilde-dot shorthand for inline functions. The notation works by replacing

```
# R code
function(x) {
  x + 10
}
```

```
## function(x) {
##   x + 10
## }
```

with

```
# R code
~{.x + 10}
```

```
## ~{
##     .x + 10
## }
```

~ indicates that you have started an anonymous function, and the argument of the anonymous function can be referred to using .x (or simply .). Unlike normal function arguments that can be anything that you like, the tilde-dot function argument is always .x.

For instance, to identify how many missing values there are in every column, we could specify the inline function ~sum(is.na(.)), which calculates how many NA values are in each column (where the column is represented by .) and adds them up:

```
# R code
dat<-data.frame(a=c(1,2,3,NA,NA,6),b=1:6,d=c(NA,2:6))
dat
```

```
##    a b  d
## 1  1 1 NA
## 2  2 2  2
## 3  3 3  3
## 4 NA 4  4
## 5 NA 5  5
## 6  6 6  6
```

```
dat %>%
  summarise(across(everything(),  ~sum(is.na(.))))  %>% head()
```

```
##   a b d
## 1 2 0 1
```

```
# R code
dat<-data.frame(a=c(1:4),b=c(1:4)^2,d=c(1:4)^3)
dat  %>% head()
```

```
##   a  b  d
## 1 1  1  1
## 2 2  4  8
## 3 3  9 27
## 4 4 16 64
```

```
dat %>%
  summarise(across(everything(),  ~ .x +10))  %>% head()
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
##    a  b  d
## 1 11 11 11
## 2 12 14 18
## 3 13 19 37
## 4 14 26 74
```

Contact us

Contact me at masoudfaridi@modares.ac.ir or masoud1faridi@gmail.com