10. Exploring Data with dplyr (3)

CT1100 - J. Duggan

dplyr

- All verbs (functions) work similarly
- The first argument is a data frame/tibble
- The subsequent arguments decide what to do with the data frame
- The result is a data frame (supports chaining of steps)

Function	Purpose
filter()	Pick observations by their values
arrange()	Reorder the rows
select()	Pick variables by their names
mutate()	Create new variables with functions of existing variables
summarise()	Collapse many values down to a single summary

(5) summarise()

- The last key verb is summarise()
- It collapses a data frame into a single row
- Not very useful unless paired with group_by()
- Very useful to combine with the pipe operator %>%
- The pipe %>% comes from the magrittr package (Stefan Milton Bache)
- Helps to write code that is easier to read and understand
  - x %>% f(y) turns into f(x, y)

```
mpg %>% select(model,displ,cty) %>% slice(1:2)
```

```
## # A tibble: 2 x 3
##   model displ   cty
##   <chr> <dbl> <int>
## 1 a4      1.8    18
## 2 a4      1.8    21
```

The function group_by()

- Most summary data operations are useful done on groups defined by variables in the the dataset.
- The group_by function takes an existing tbl and converts it into a grouped tbl where operations can then performed "by group".

```
gr <- group_by(mpg,year)
agg <- summarise(gr,AverageCty=mean(cty))
agg
```

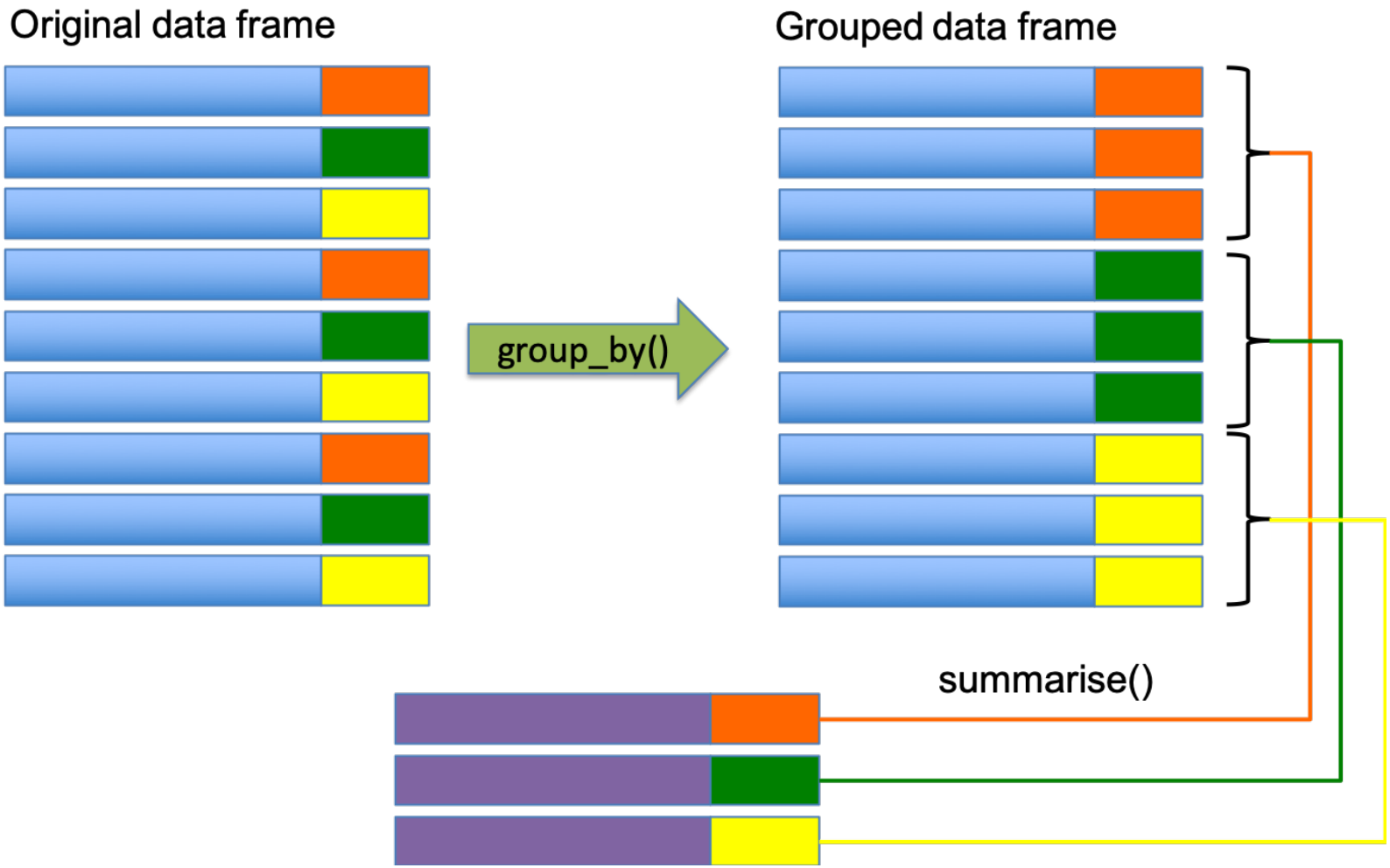
```
## # A tibble: 2 x 2
##    year AverageCty
##   <int>      <dbl>
## 1  1999       17.0
## 2  2008       16.7
```

Using %>%

```
mpg %>% group_by(manufacturer)                %>%
        summarise(AvrCty=mean(cty),N=n())     %>%
        arrange(desc(AvrCty))                 %>%
        slice(1:5)
```

```
## # A tibble: 5 x 3
##   manufacturer AvrCty     N
##   <chr>          <dbl> <int>
## 1 honda           24.4     9
## 2 volkswagen      20.9    27
## 3 subaru          19.3    14
## 4 hyundai         18.6    14
## 5 toyota          18.5    34
```

Overall idea



Useful Summary Functions

Grouping	Examples
Measures of location	mean(), median()
Measures of spread	sd(), IQR(),mad()
Measures of rank	min((), quantile(), max()
Measures of position	first(), nth(), last()
Counts	n(), n_distinct()
Counts and proportions of logical values	sum(x>0) when used with numeric functions, (T,F) converted to (1,0)

The package nycflights13

```
glimpse(nycflights13::flights)
```

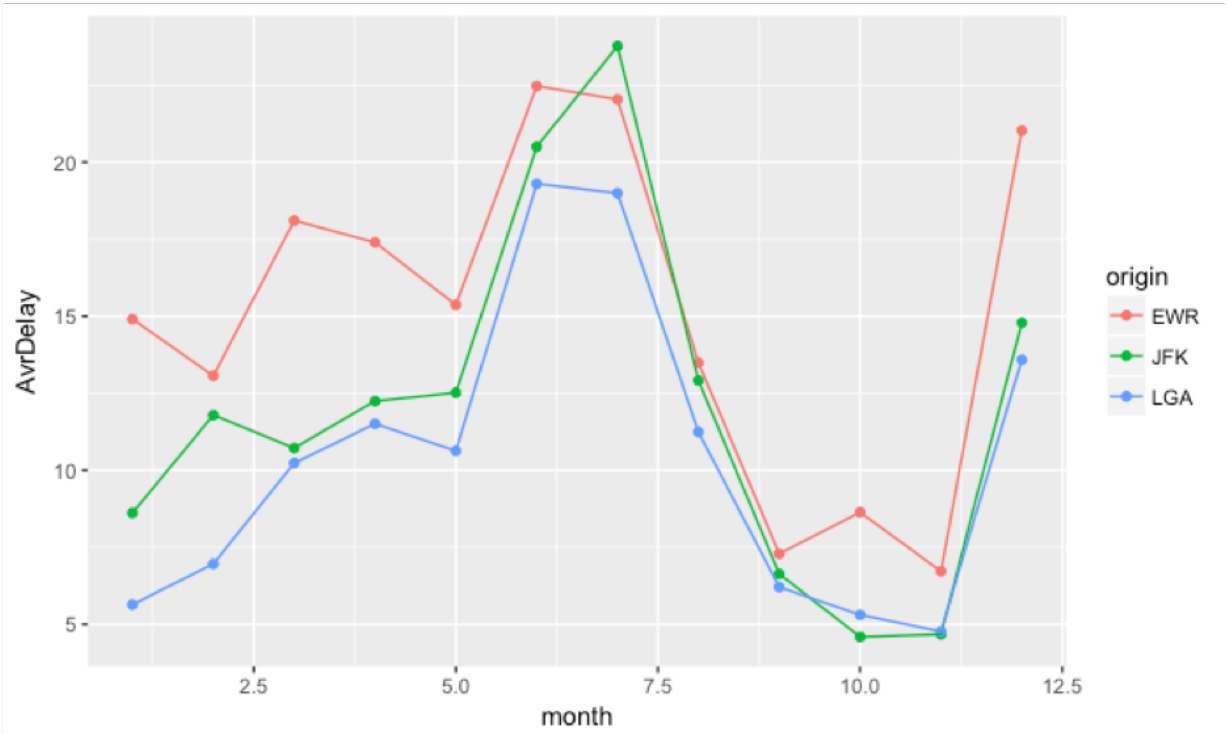
```
## Observations: 336,776
## Variables: 19
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013,
## $ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
## $ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
## $ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 5
## $ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 6
## $ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2,
## $ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913,
## $ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854,
## $ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -
## $ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA",
## $ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 50
## $ tailnum        <chr> "N14228", "N24211", "N619AA", "N804J
```

Challenge 2.2 | nycflights13::flights

Generate the following graph. Use the variable **dep_delay**. The variable **origin** indicates the departure airport.

```
unique(nycflights13::flights$origin)
```

```
## [1] "EWR" "LGA" "JFK"
```



Overall Summary

- dplyr - a grammar of data manipulation
- Five verbs
  - **filter()**
  - **arrange()**
  - **select()**
  - **mutate()**
  - **summarise()** (along with **group_by()**)
- Usefully combined with **%>%** operator