p8105_hw1_al4771

Anqi Li

2025-09-19

```r
library(conflicted)
conflicts_prefer(dplyr::filter)
```

```
## [conflicted] Will prefer dplyr::filter over any other package.
```

```r
# Import the 'tidyverse' library
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.1.0
```

```r
# Import the 'moderndive' library
library(moderndive)

# Load the 'early_january_weather' dataset
data("early_january_weather")
```

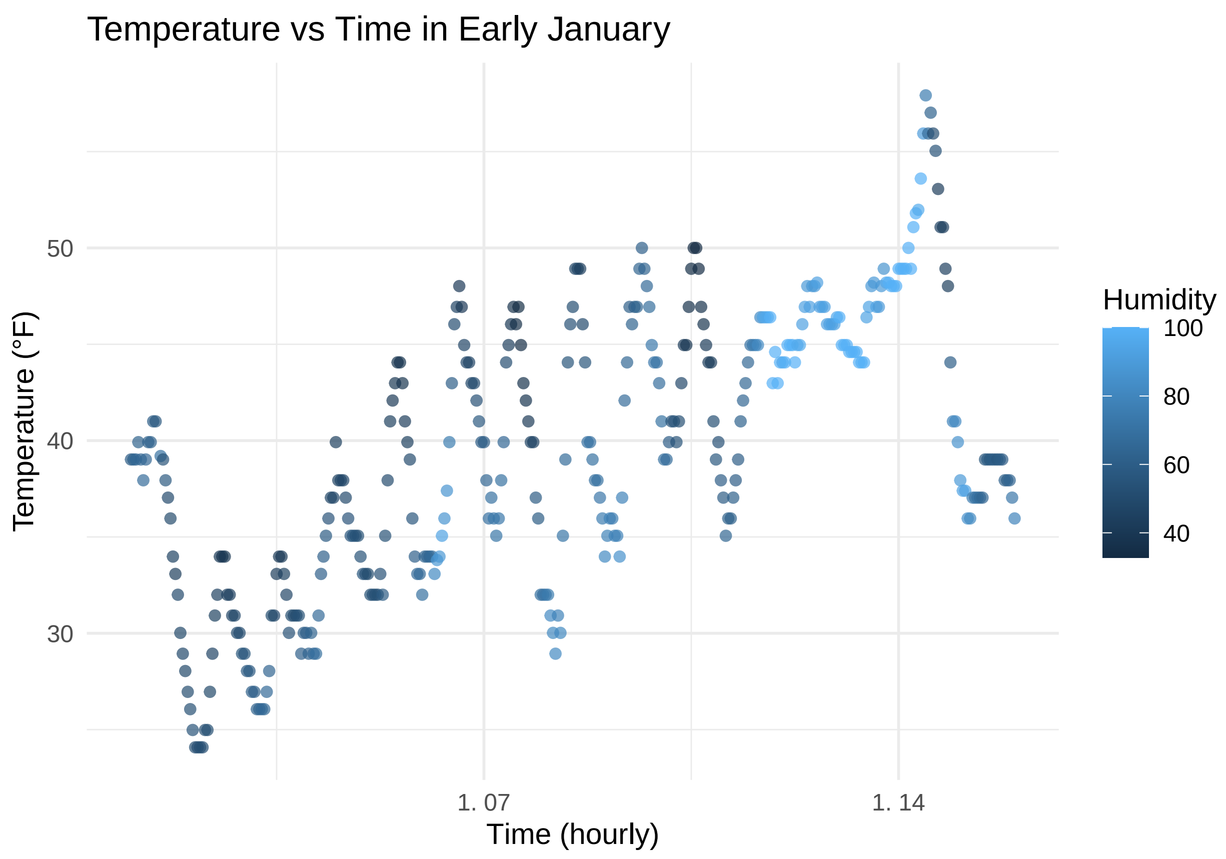
Problem 1

The variables in this dataset are **origin, year, month, day, hour, temp, dewp, humid, wind_dir, wind_speed, wind_gust, precip, pressure, visib, time_hour**.

The dataset **early_january_weather** contains hourly weather data from **January 2013** in **New York City**. The dataset has **358** rows and and **15** columns. The average temperature across this dataset is about **39.6** °F.

```r
weather_plot <- ggplot(
  early_january_weather,
  aes(x = time_hour,
      y = temp,
      color = humid)
  ) +
  geom_point(alpha = 0.7) +
  labs(
    title = "Temperature vs Time in Early January",
    x = "Time (hourly)",
    y = "Temperature (°F)",
    color = "Humidity"
  ) +
  theme_minimal()

weather_plot
```



```
# Export the scatterplot in PNG form
ggsave("scatterplot_weather.png",
       plot = weather_plot,
       width = 6,
       height = 4)
```

Problem 2

```
# For reproducibility
set.seed(123)

# Create a dataframe
# numeric: random sample
x_num <- rnorm(10)

# logical: indicator whether values > 0
x_logical <- x_num > 0

# character: vector of length 10
```

```r
x_char <- paste0("lab", 1:10)

# factor: 3 different levels
x_factor <- factor(rep(c("low", "medium", "high"),
                       length.out = 10))

# combine
df <- tibble(x_num, x_logical, x_char, x_factor)
df
```

```
## # A tibble: 10 x 4
##      x_num x_logical x_char x_factor
##      <dbl> <lgl>     <chr>  <fct>
##  1 -0.560  FALSE     lab1   low
##  2 -0.230  FALSE     lab2   medium
##  3  1.56   TRUE      lab3   high
##  4  0.0705 TRUE      lab4   low
##  5  0.129  TRUE      lab5   medium
##  6  1.72   TRUE      lab6   high
##  7  0.461  TRUE      lab7   low
##  8 -1.27   FALSE     lab8   medium
##  9 -0.687  FALSE     lab9   high
## 10 -0.446  FALSE     lab10  low
```

```r
# Take the mean of each variable in my dataframe
mean(pull(df, x_num))
```

```
## [1] 0.07462564
```

```r
mean(pull(df, x_logical))
```

```
## [1] 0.5
```

```r
mean(pull(df, x_char))   # will fail
```

```
## Warning in mean.default(pull(df, x_char)): argument is not numeric or logical:
## returning NA

## [1] NA
```

```r
mean(pull(df, x_factor)) # will fail
```

```
## Warning in mean.default(pull(df, x_factor)): argument is not numeric or
## logical: returning NA

## [1] NA
```

```r
# Convert variables from one type to another
# TRUE/FALSE -> 1/0
as.numeric(df$x_logical) # TRUE/FALSE -> 1/0
```

```
##  [1] 0 0 1 1 1 1 1 0 0 0
```

```r
as.numeric(df$x_char) # characters -> NA with warning
```

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

```
##  [1] NA NA NA NA NA NA NA NA NA NA
```

```r
as.numeric(df$x_factor) # factors -> underlying integer codes
```

```
##  [1] 2 3 1 2 3 1 2 3 1 2
```

The mean works for numeric and logical variables but not for character or factor variables. This is because only numeric (and logical, due to coercion) are treated as quantitative data. Characters do not have numeric meaning, and factors are categorical variables whose integer encodings should not be interpreted as numbers.