Tidyverse Create Asignment

Noori Selina

2023-11-07

```r
# Load necessary libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Introduction

This vignette demonstrates how to perform data manipulation and analysis using the dplyr package. We will work with the Census Income dataset, which contains information on individuals' demographics and income levels.The data set can be found here: https://www.kaggle.com/datasets/tawfikelmetwally/census-income-dataset/data

Data Loading

To load the data, I will be importing the data from a GitHub link.

```r
census_data <- read_csv("https://raw.githubusercontent.com/NooriSelina/Data-607/main/censusincome.csv")
```

```
## Rows: 32561 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (9): Workclass, Education, Marital Status, Occupation, Relationship, Rac...
## dbl (6): Age, Final Weight, EducationNum, Capital Gain, capital loss, Hours ...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(census_data)
```

```
## # A tibble: 6 x 15
##     Age Workclass        ‘Final Weight‘ Education EducationNum ‘Marital Status‘
##   <dbl> <chr>                     <dbl> <chr>            <dbl> <chr>
## 1    39 State-gov                 77516 Bachelors           13 Never-married
## 2    50 Self-emp-not-inc          83311 Bachelors           13 Married-civ-spou~
## 3    38 Private                  215646 HS-grad              9 Divorced
## 4    53 Private                  234721 11th                 7 Married-civ-spou~
## 5    28 Private                  338409 Bachelors           13 Married-civ-spou~
## 6    37 Private                  284582 Masters             14 Married-civ-spou~
## # i 9 more variables: Occupation <chr>, Relationship <chr>, Race <chr>,
## #   Gender <chr>, ‘Capital Gain‘ <dbl>, ‘capital loss‘ <dbl>,
## #   ‘Hours per Week‘ <dbl>, ‘Native Country‘ <chr>, Income <chr>
```

Data Manipulation with dplyr

The dplyr package provides a set of functions for data manipulation. We will use the following functions to explore the dataset.

1. Filtering - Using the dplyr package, I will use the `filter()` function to filter the dataset based on specific criteria. In this case, we are interested in individuals with incomes greater than $50,000, so we filter the dataset to include only such individuals.

```
high_income_data <- census_data %>%
  filter(Income == ">50K")
head(high_income_data)
```

```
## # A tibble: 6 x 15
##     Age Workclass        ‘Final Weight‘ Education  EducationNum ‘Marital Status‘
##   <dbl> <chr>                     <dbl> <chr>             <dbl> <chr>
## 1    52 Self-emp-not-inc         209642 HS-grad               9 Married-civ-spo~
## 2    31 Private                   45781 Masters              14 Never-married
## 3    42 Private                  159449 Bachelors            13 Married-civ-spo~
## 4    37 Private                  280464 Some-coll~           10 Married-civ-spo~
## 5    30 State-gov                141297 Bachelors            13 Married-civ-spo~
## 6    40 Private                  121772 Assoc-voc            11 Married-civ-spo~
## # i 9 more variables: Occupation <chr>, Relationship <chr>, Race <chr>,
## #   Gender <chr>, ‘Capital Gain‘ <dbl>, ‘capital loss‘ <dbl>,
## #   ‘Hours per Week‘ <dbl>, ‘Native Country‘ <chr>, Income <chr>
```

2. Grouping and Summarizing - The group_by() and summarize() functions of the dplyr package are valuable for aggregating data. We will group the data by education level and calculate summary statistics for age and hours worked per week within each education category.

```
income_summary <- high_income_data %>%
  group_by(Income) %>%
  summarize(
    mean_age = mean(Age, na.rm = TRUE),
    median_hours = median(`Hours per Week`, na.rm = TRUE)
  )

print(income_summary)
```

```
## # A tibble: 1 x 3
##   Income mean_age median_hours
##   <chr>     <dbl>        <dbl>
## 1 >50K       44.2           40
```

3. Sorting - The arrange() function is used to sort the summarized data. In this example, we sort the education summary by mean age in descending order, which helps identify the education categories with the highest mean age.

```
high_income_data <- high_income_data %>%
  arrange(desc(Age))

print(head(high_income_data, 10))
```

```
## # A tibble: 10 x 15
##      Age Workclass    ‘Final Weight‘ Education    EducationNum ‘Marital Status‘
##    <dbl> <chr>                 <dbl> <chr>               <dbl> <chr>
##  1    90 Local-gov            227796 Masters                14 Married-civ-spou~
##  2    90 Private               51744 Masters                14 Never-married
##  3    90 Private               87372 Prof-school            15 Married-civ-spou~
##  4    90 Private               46786 Bachelors              13 Married-civ-spou~
##  5    90 Private              175491 HS-grad                 9 Married-civ-spou~
##  6    90 Private               88991 Bachelors              13 Married-civ-spou~
##  7    90 Private              206667 Masters                14 Married-civ-spou~
##  8    90 ?                    313986 HS-grad                 9 Married-civ-spou~
##  9    84 Self-emp-inc         172907 Some-college           10 Married-civ-spou~
## 10    83 Self-emp-inc         240150 10th                    6 Married-civ-spou~
## # i 9 more variables: Occupation <chr>, Relationship <chr>, Race <chr>,
## #   Gender <chr>, ‘Capital Gain‘ <dbl>, ‘capital loss‘ <dbl>,
## #   ‘Hours per Week‘ <dbl>, ‘Native Country‘ <chr>, Income <chr>
```

Conclusion

In this vignette, the dplyr package from the tidyverse collection was used to efficiently manage and analyze the Census Income dataset. Specifically, the data was filtered using the filter() function to focus on high-income individuals, grouped by income levels using the group_by() function, and then sorted by mean age with the arrange() function.

This approach uncovered valuable insights about the demographics of high-income individuals. The dplyr package, along with other tools from the tidyverse, made data manipulation tasks straightforward and effective.