CUSTOMER ANALYTICS :PREPARING DATA FOR MODELLING

2024-06-02

Instructions: The Head Data Scientist at Training Data Ltd. has asked you to create a data frame called ds_jobs_clean that stores the data in customer_train.csv much more efficiently. Specifically, they have set the following requirements:

Columns containing integer values must be stored as integers (). Columns containing float values must be stored as doubles (). Columns containing categorical data, both ordinal and nominal, must be stored as factors (). Collapse the company_size levels to the following: 'Micro': <10 employees 'Small': 10-99 employees 'Medium': 100-999 employees 'Large': >1000 employees Collapse the experience levels to the following: '<5': <5 year of experience '5-10': 5-10 years of experience '>10': >10 years of experience The columns of ds_jobs_clean must be in the same order as the original dataset. The data frame should be filtered to only contain students with 10 or more years of experience at companies with at least 1000 employees, as their recruiter base is suited to more experienced professionals at large enterprise companies. If you call object.size() on ds_jobs and ds_jobs_clean after you've preprocessed it, you should notice a substantial decrease in memory usage.

1. Exploratory data analysis

Load customer_train.csv to begin exploring the data to understand the contents and data types of the values in each column. Loading a CSV file You can use the read_csv() function from the readr library to load the CSV file as a tibble. How to find the data types and contents of a column You can check the column names and assigned data types by calling the glimpse() function on the tibble. To view the unique values and counts present in a column, pipe the tibble into count() from dplyr, passing the function the variable whose values you wish to count.

```
library(readr)

## Warning: package 'readr' was built under R version 4.3.2

library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(forcats)
setwd("C:/Users/EE User/OneDrive/Gopika/OMSA/MGT-6203/r programing")
# Load the dataset
ds_jobs <- read_csv("customer_train.csv",show_col_types = FALSE)
head(ds_jobs,5)

## # A tibble: 5 × 14
##   student_id city     city_development_index gender relevant_experience
##        <dbl> <chr>                     <dbl> <chr>  <chr>
## 1       8949 city_103                  0.92  Male   Has relevant
experience
## 2      29725 city_40                   0.776 Male   No relevant experience
## 3      11561 city_21                   0.624 <NA>   No relevant experience
## 4      33241 city_115                  0.789 <NA>   No relevant experience
## 5        666 city_162                  0.767 Male   Has relevant
experience
## # ℹ 9 more variables: enrolled_university <chr>, education_level <chr>,
## #   major_discipline <chr>, experience <chr>, company_size <chr>,
## #   company_type <chr>, last_new_job <chr>, training_hours <dbl>,
## #   job_change <dbl>
```

2. Converting column types

Convert columns containing integers to the type, floats to the type, and categorical data to the type. Functions for converting column types The as.integer(), as.numeric(), and as.factor() functions can be used to convert data types, including data frame columns. Applying functions to certain columns The mutate() and across() combination can be used to apply functions to particular columns. The syntax for mutating across columns: mutate(across(,)). Columns can be selected using a vector, or using a tidyselect helper function like where() to select on a condition. Collapse factor levels to reduce the amount of unnecessary information being stored. The fct_collapse() function inside mutate() can be used to redefine the levels used in a factor column. The syntax for collapsing columns: mutate(= fct_collapse(, = c(), ...)))

```
ds_jobs_clean <- ds_jobs %>%
  # Convert student_id, training_hours, and job_change to integers
  mutate(across(c(student_id, training_hours, job_change), as.integer)) %>%
  # Convert city_development_index to a numeric
  mutate(across(city_development_index, as.numeric)) %>%
  # Convert the remaining character columns to factors
  mutate(across(where(is.character), as.factor)) %>%
  # Collapse company_size levels to Micro, Small, Medium, and Large
  mutate(company_size = fct_collapse(company_size,
        'Micro' = '<10',
        'Small' = c('10-49', '50-99'),
        'Medium' = c('100-499', '500-999'),
        'Large' = c('1000-4999', '5000-9999', '10000+'))) %>%
  # Collapse experience levels to <5, 5-10, >10
  mutate(experience = fct_collapse(experience,
        '<5' = c('<1', as.character(1:4)),
```

```r
       '5-10' = as.character(5:10),
       '>10' = c(as.character(11:20), '>20'))) %>%
  # Filter students with >10 years experience from large enterprises
  filter(company_size == 'Large', experience == '>10')

ds_jobs_clean
```

```
## # A tibble: 1,956 × 14
##    student_id city     city_development_index gender relevant_experience
##         <int> <fct>                     <dbl> <fct>  <fct>
##  1        699 city_103                  0.92  <NA>   Has relevant
experience
##  2      25619 city_61                   0.913 Male   Has relevant
experience
##  3      22293 city_103                  0.92  Male   Has relevant
experience
##  4      26494 city_16                   0.91  Male   Has relevant
experience
##  5       2547 city_114                  0.926 Female Has relevant
experience
##  6      25987 city_103                  0.92  Other  Has relevant
experience
##  7       1180 city_16                   0.91  Male   Has relevant
experience
##  8      25349 city_16                   0.91  Male   Has relevant
experience
##  9      20576 city_97                   0.925 Male   Has relevant
experience
## 10       3921 city_36                   0.893 <NA>   No relevant
experience
## # ℹ 1,946 more rows
## # ℹ 9 more variables: enrolled_university <fct>, education_level <fct>,
## #   major_discipline <fct>, experience <fct>, company_size <fct>,
## #   company_type <fct>, last_new_job <fct>, training_hours <int>,
## #   job_change <int>
```