salary_prediction

Masahiro TAKAOKA

2020/1/8

Overview

This project is part of the HarvardX: PH125.9x Data Science: Capstone course and the purpose of this project is to create an analysis report using a dataset of our own choosing.

Salary prediction was held at a data analysis competition called "prob.space" from November 6, 2019 to December 23, 2019. This site is a data analysis competition that was released in 2019 and can be called the Japanese version of kaggle, which is still in beta. This site is premised on participation using Python, but for the final report of the PH125.9x Data Science, I used the data set of this competition and analyzed using R.

Note: If the result is not returned due to heavy processing, please run again or change the equipment.

Competition Overview

Akaike-kun, who is in charge of personnel at a company, has lost the salary rules for determining employee salaries. At the same time, the bankbook was lost and some employees lost their salary information.

Employee information and salary data will be given, so let's predict the salary information of the lost employee to help Akaike who is in trouble.

Dataset description

The data is divided into two groups.

Training data (train_data.csv) Test data (test_data.csv)

The training data set and test data set contain 21,000 and 9,000 samples, respectively. Use the training data set to build a machine learning model. For the training data set, salary values are given. Since we need to use a test dataset to see how much the model has been run against new data, There is no salary data for each employee. So use test data sets to predict each employee's salary.

Please refer to the tag of evaluation method for an example of the submitted file.

Data format

The columns of the data set are as follows. The original data set used in this competition had Japanese prefecture names stored in the area column. In order to be reviewed in the final assignment of this course, the area column has been replaced with the prefecture code according to the rules of Japanese ISO.

position : position (0 = No position, 1 = Chief, 2 = Senior Staff, 3 = Manager, 4 = General Manager)
age : age
area : Prefecture code (https://en.wikipedia.org/wiki/Prefectures_of_Japan#By_Japanese_ISO)
sex : gender (1 = male, 2 = female)
partner presence / absence of spouse : (0 = none, 1 = present)
num_child : Number of children (people)
education : (0 = high school, 1 = Associate degree, 2 = Bachelor, 3 = master, 4 = doctor)
service_length : Length of service (years)
study_time : Study time per week (h)

commute : commute time (h)
overtime : Overtime hours per month (h)

salary : monthly salary (in units of 1,000 yen)

Evaluation

Goal

The goal is to predict the employee's salary based on the employee data in the test data set. Predict y (salary) with a real value.

metric

The predictive performance of the model is evaluated by MAE (mean absolute error) of predicted values and true values from 9,000 test data sets.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y - \hat{y}|$$

Here y is the true value, $\hat{y}$ is the predicted value.

Submission file format

Prepare a submission submission.csv (including 9,000 entries and header lines). An error will occur if the submitted file contains extra rows or columns (other than id and y).

The submission file must contain only the following columns:

id (same order as test data set)
y (predicted salary)

Install required packages

```
# Install and load required packages
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")
# if(!require(readxl)) install.packages("readxl", repos = "http://cran.us.r-project.org")
if(!require(readr)) install.packages("readr", repos = "http://cran.us.r-project.org")
if(!require(psych)) install.packages("psych", repos = "http://cran.us.r-project.org")
if(!require(RcppEigen)) install.packages("RcppEigen", repos = "http://cran.us.r-project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")
if(!require(ranger)) install.packages("ranger", repos = "http://cran.us.r-project.org")
library(caret)
library(ggplot2)
library(dplyr)
library(readr)
library(psych)
library(RcppEigen)
library(e1071)
library(ranger)
```

Loading a dataset (Prepared on Github)

```
# https://github.com/TAKAOKA-Masahiro/salary_prediction

# Data type is specified
test_data_eng <- read_csv("test_data_eng.csv", col_types=("iiiiiiiiiidd"))
train_data_eng <- read_csv("train_data_eng.csv", col_types=("iiiiiiiiiiddd"))
```

Data exploration and visualization

First, check the data structure and basic statistics. The target variable is salary, and a model that predicts this salary is created using other variables.

```
# Check the first few lines
head(train_data_eng)
```

```
## # A tibble: 6 x 13
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        1    44    23     2       1         2         1
## 2     1        2    31    29     1       0         0         0
## 3     2        2    36    35     1       0         0         2
## 4     3        0    22    13     2       0         0         0
## 5     4        0    25    46     2       0         0         1
## 6     5        1    23    28     2       1         3         1
## # ... with 5 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, salary <dbl>
```

```
# Check the data structure
str(train_data_eng)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 21000 obs. of  13 variables:
##  $ id            : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ position      : int  1 2 2 0 0 1 1 0 3 2 ...
##  $ age           : int  44 31 36 22 25 23 32 21 43 46 ...
##  $ area          : int  23 29 35 13 46 28 14 29 4 8 ...
##  $ sex           : int  2 1 1 2 2 2 2 1 2 2 ...
##  $ partner       : int  1 0 0 0 0 1 1 0 0 1 ...
##  $ num_child     : int  2 0 0 0 0 3 2 0 0 2 ...
##  $ education     : int  1 0 2 0 1 1 0 0 0 2 ...
##  $ service_length: int  24 13 14 4 5 3 14 3 25 24 ...
##  $ study_time    : int  2 9 4 3 3 2 5 3 11 7 ...
##  $ commute       : num  1.6 0.7 0.4 0.4 0.2 1.3 1.7 0.1 0.7 1.7 ...
##  $ overtime      : num  9.2 12.4 16.9 6.1 4.9 7.4 11.8 18.9 10.1 4.8 ...
##  $ salary        : num  428 318 357 201 178 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_integer(),
##   ..   position = col_integer(),
##   ..   age = col_integer(),
##   ..   area = col_integer(),
##   ..   sex = col_integer(),
##   ..   partner = col_integer(),
##   ..   num_child = col_integer(),
```

```
##   ..   education = col_integer(),
##   ..   service_length = col_integer(),
##   ..   study_time = col_integer(),
##   ..   commute = col_double(),
##   ..   overtime = col_double(),
##   ..   salary = col_double()
##   .. )
```

```
# Check basic statistics
summary(train_data_eng)
```

```
##        id           position          age             area
##  Min.   :    0   Min.   :0.000   Min.   :18.00   Min.   : 1.0
##  1st Qu.: 5250   1st Qu.:0.000   1st Qu.:24.00   1st Qu.:12.0
##  Median :10500   Median :1.000   Median :30.00   Median :24.0
##  Mean   :10500   Mean   :1.227   Mean   :33.13   Mean   :24.2
##  3rd Qu.:15749   3rd Qu.:2.000   3rd Qu.:42.00   3rd Qu.:36.0
##  Max.   :20999   Max.   :4.000   Max.   :67.00   Max.   :47.0
##       sex           partner         num_child        education
##  Min.   :1.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :1.000   Median :0.0000   Median :0.0000   Median :1.000
##  Mean   :1.498   Mean   :0.4993   Mean   :0.9997   Mean   :1.099
##  3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:2.0000   3rd Qu.:2.000
##  Max.   :2.000   Max.   :1.0000   Max.   :9.0000   Max.   :4.000
##  service_length   study_time       commute        overtime
##  Min.   : 0.0   Min.   : 0.000   Min.   :0.10   Min.   : 0.00
##  1st Qu.: 3.0   1st Qu.: 1.000   1st Qu.:0.50   1st Qu.: 8.30
##  Median : 9.0   Median : 3.000   Median :1.10   Median :12.10
##  Mean   :12.3   Mean   : 3.828   Mean   :1.06   Mean   :12.13
##  3rd Qu.:21.0   3rd Qu.: 6.000   3rd Qu.:1.50   3rd Qu.:15.80
##  Max.   :49.0   Max.   :24.000   Max.   :4.80   Max.   :31.90
##      salary
##  Min.   : 110.6
##  1st Qu.: 225.5
##  Median : 315.2
##  Mean   : 361.2
##  3rd Qu.: 456.9
##  Max.   :1098.9
```

check null

```
# check null
sum(is.na(train_data_eng))
```

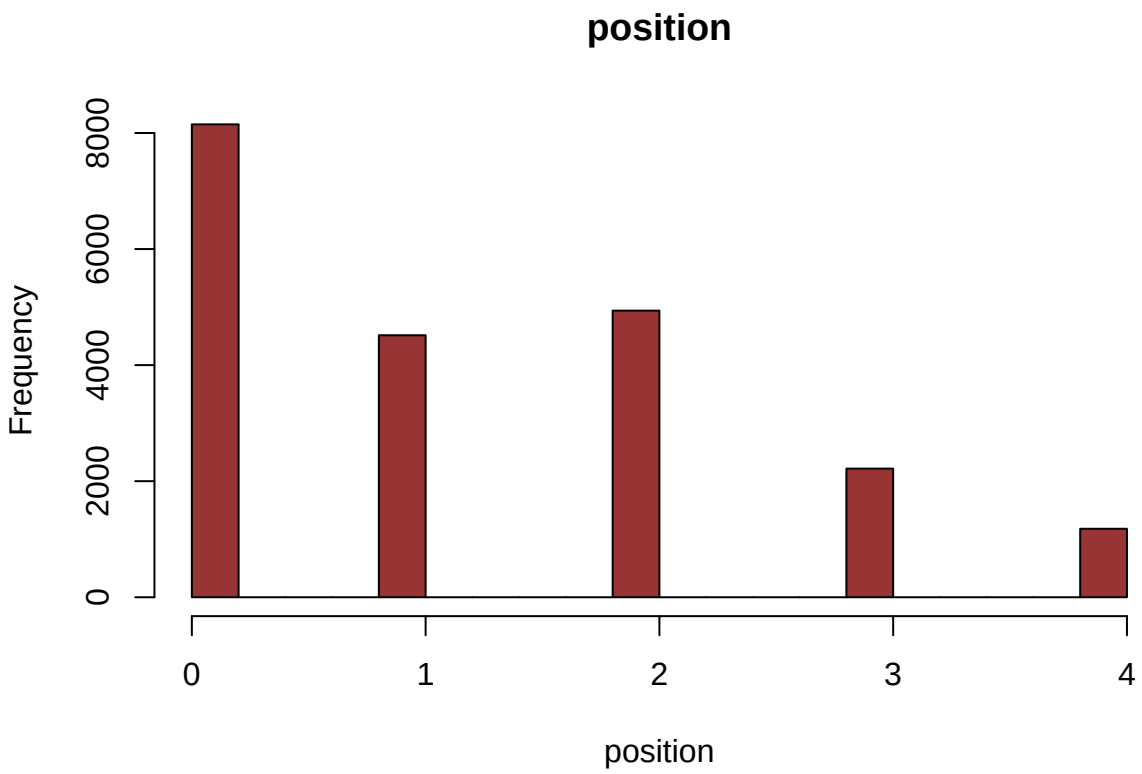
```
## [1] 0
```

```
sum(is.na(test_data_eng))
```

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

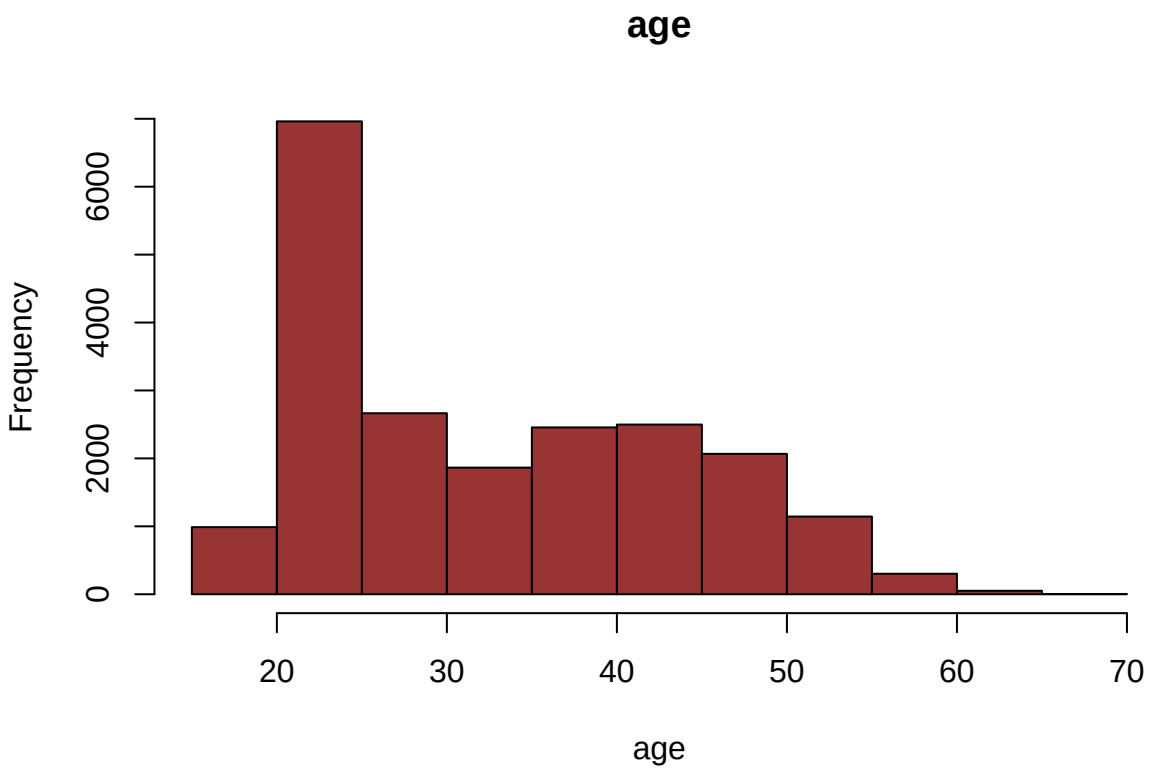
Check the distribution of data for each variable in the histogram

```
# Histogram for position
hist(train_data_eng$position, main="position", xlab="position", col="#993435")
```



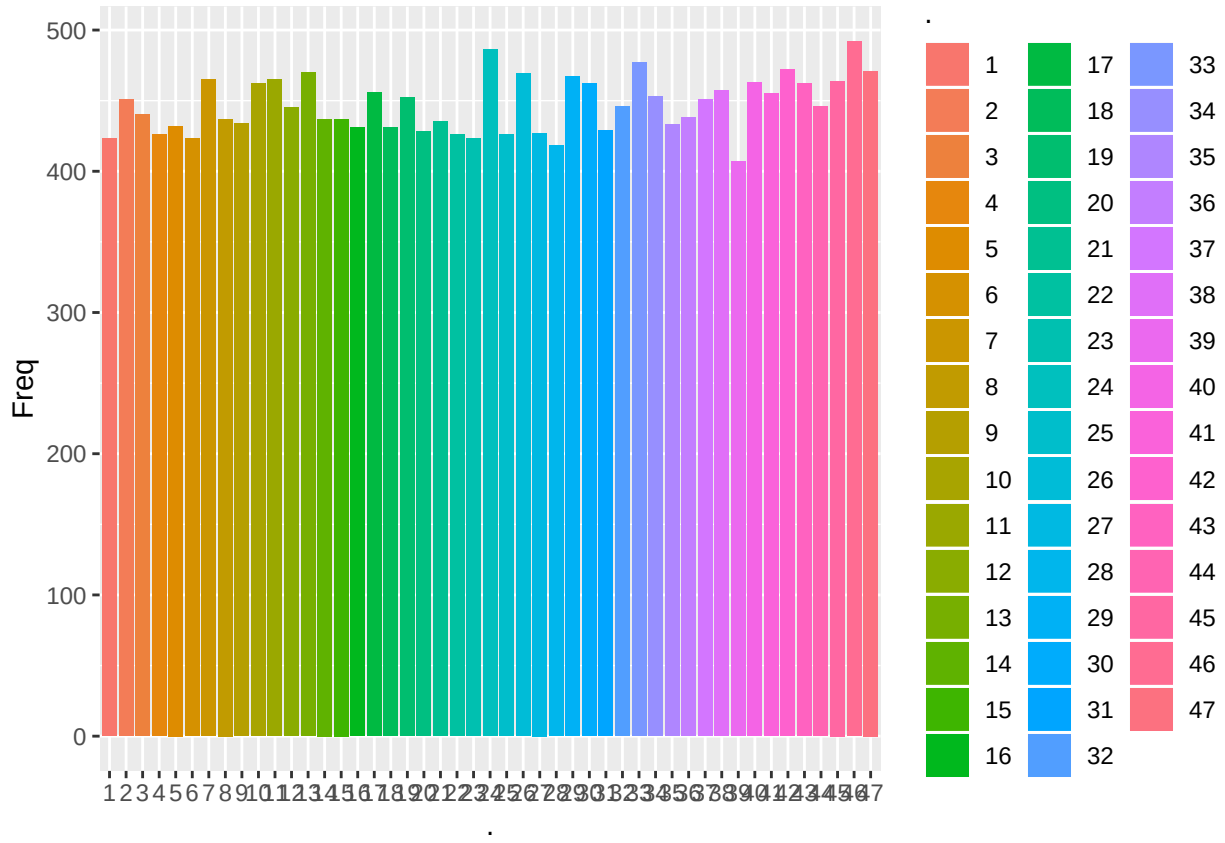
This column seems to be related to age.

```
# Histogram for age
hist(train_data_eng$age, main="age", xlab="age", col="#993435")
```



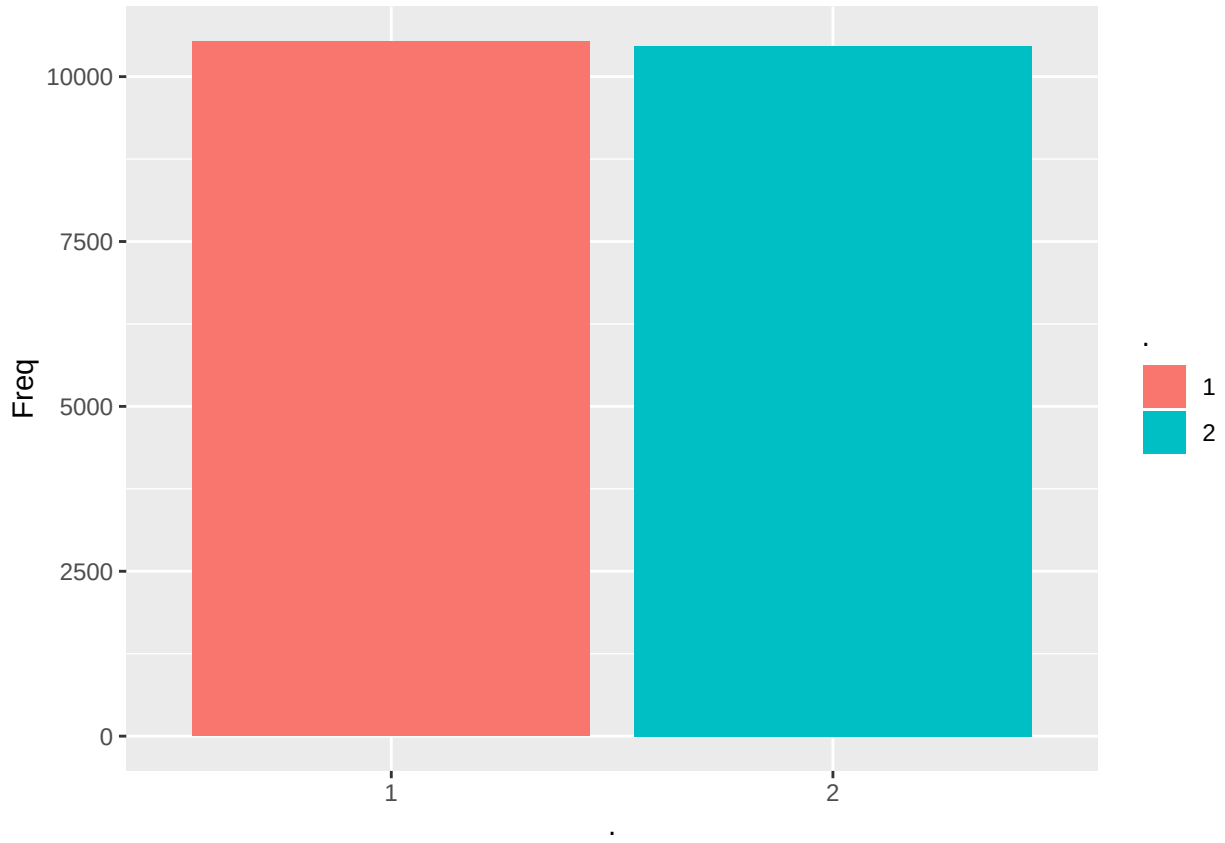
There are many records in the early 20's.

```
# Histogram for area
train_data_eng$area %>%
  table %>%
  as.data.frame() %>%
  ggplot(aes(x =  ., y = Freq, fill = .)) +
  geom_bar(stat = "identity")
```



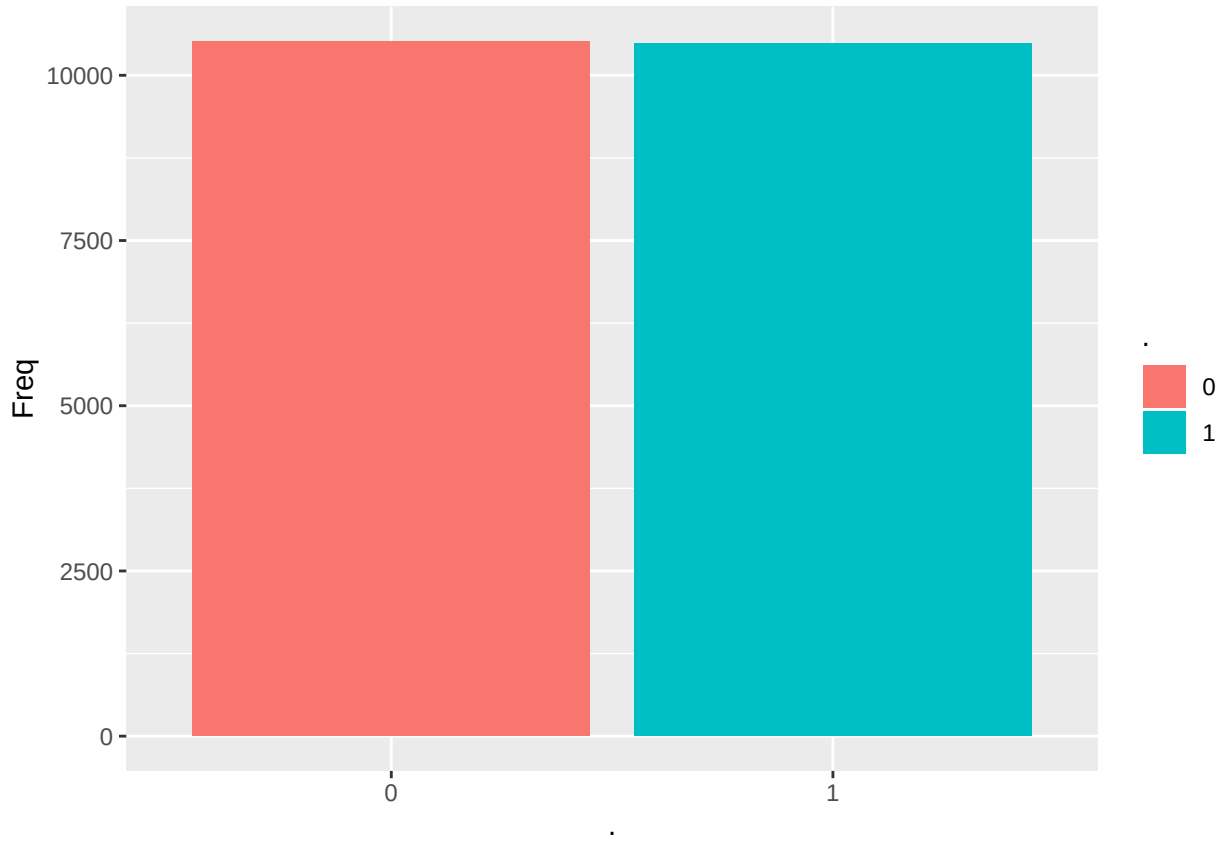
Area data is distributed evenly regardless of population.

```
# Histogram for sex
train_data_eng$sex %>%
  table %>%
  as.data.frame() %>%
  ggplot(aes(x =  ., y = Freq, fill = .)) +
  geom_bar(stat = "identity")
```



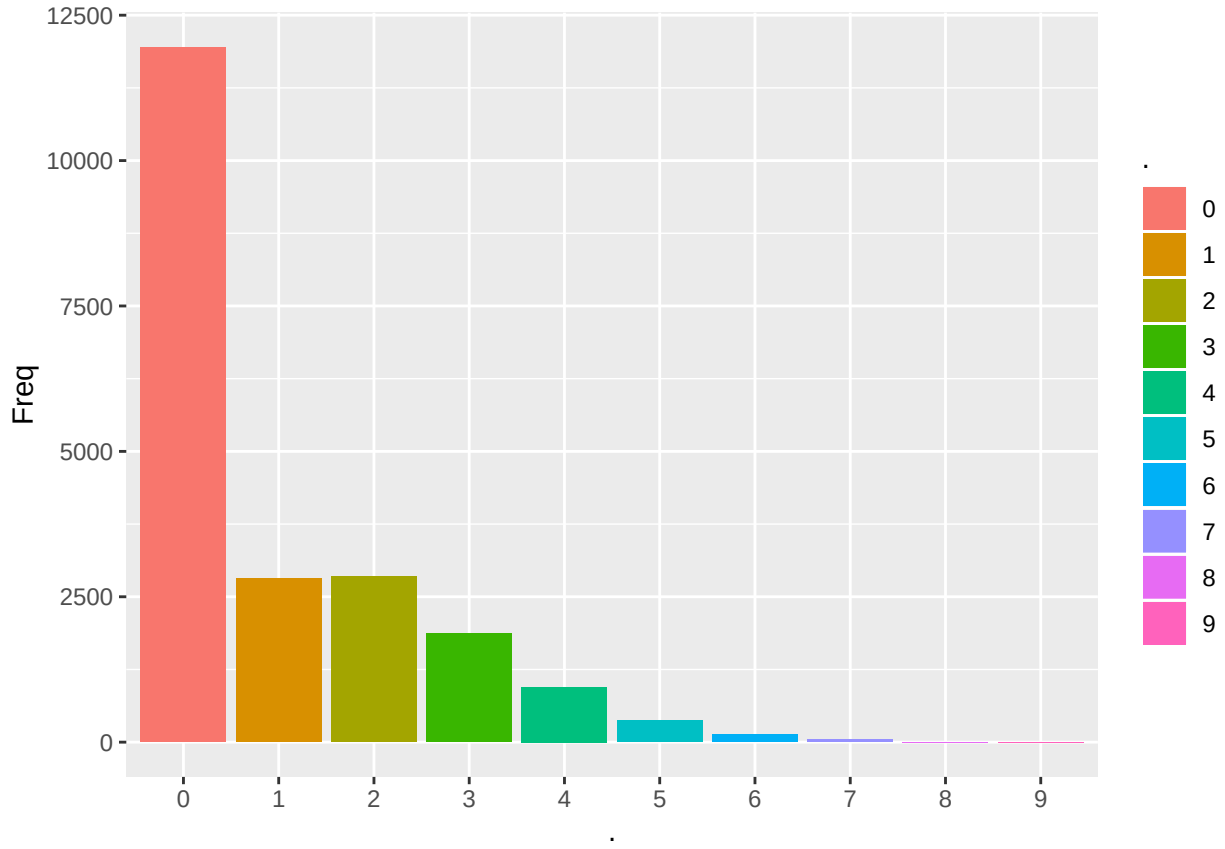
There is no difference in the number of records between genders.

```
# Histogram for partner
train_data_eng$partner %>%
  table %>%
  as.data.frame() %>%
  ggplot(aes(x =  ., y = Freq, fill = .)) +
  geom_bar(stat = "identity")
```



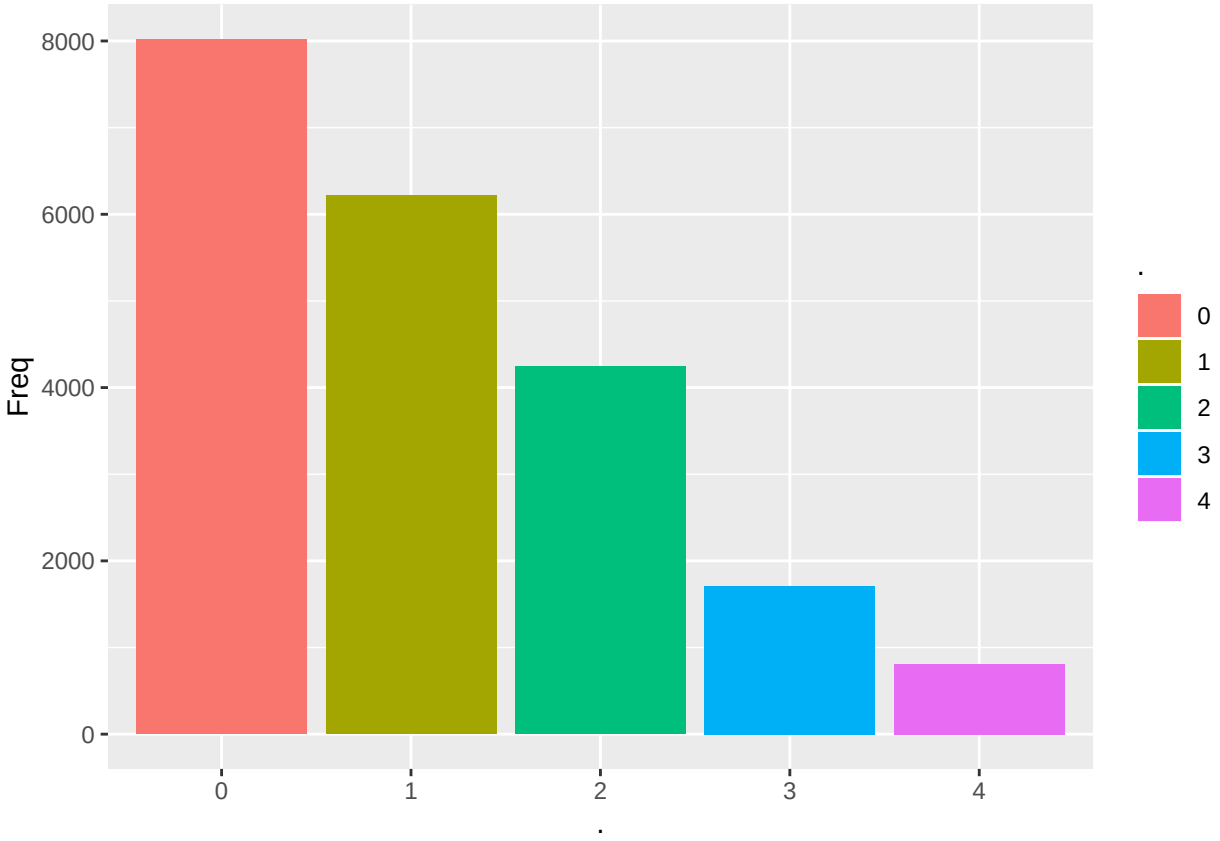
There is no difference in the number of records due to marriage status.

```
# Histogram for num_child
train_data_eng$num_child %>%
  table %>%
  as.data.frame() %>%
  ggplot(aes(x =  ., y = Freq, fill = .)) +
  geom_bar(stat = "identity")
```



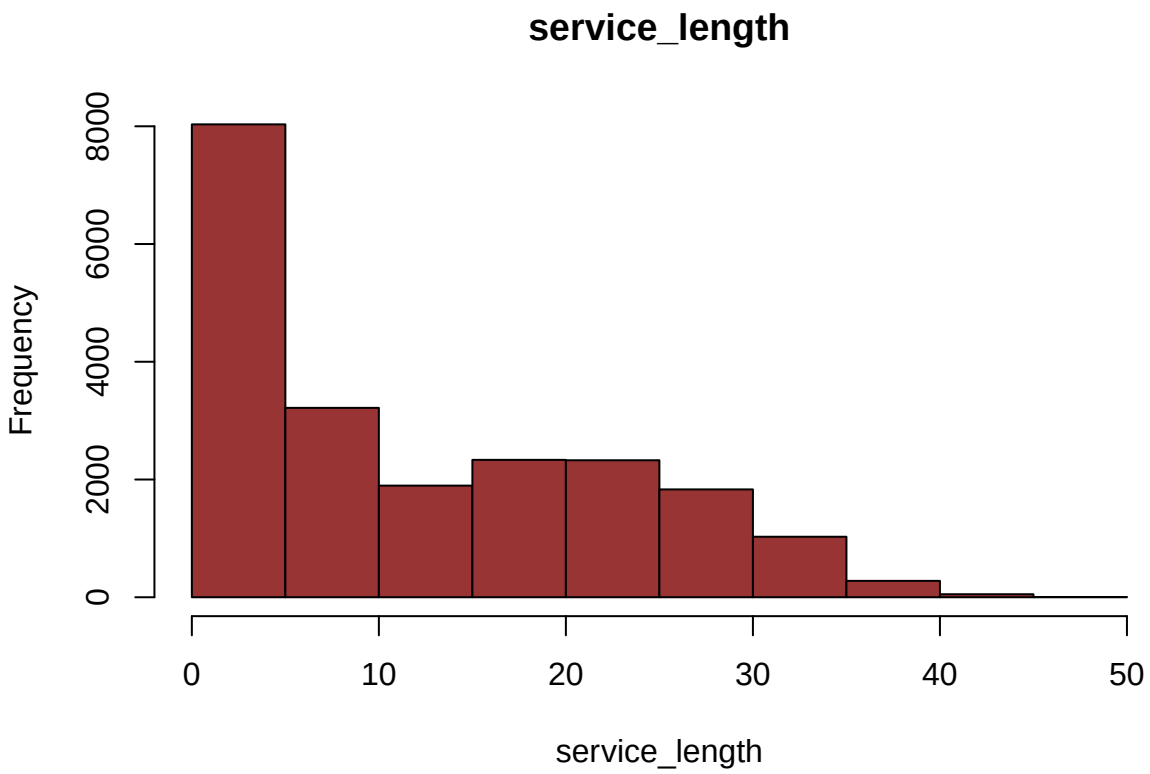
Most data is 0, but it is expected that there is a strong correlation with the partner column.

```
# Histogram for education
train_data_eng$education %>%
  table %>%
  as.data.frame() %>%
  ggplot(aes(x =  ., y = Freq, fill = .)) +
  geom_bar(stat = "identity")
```



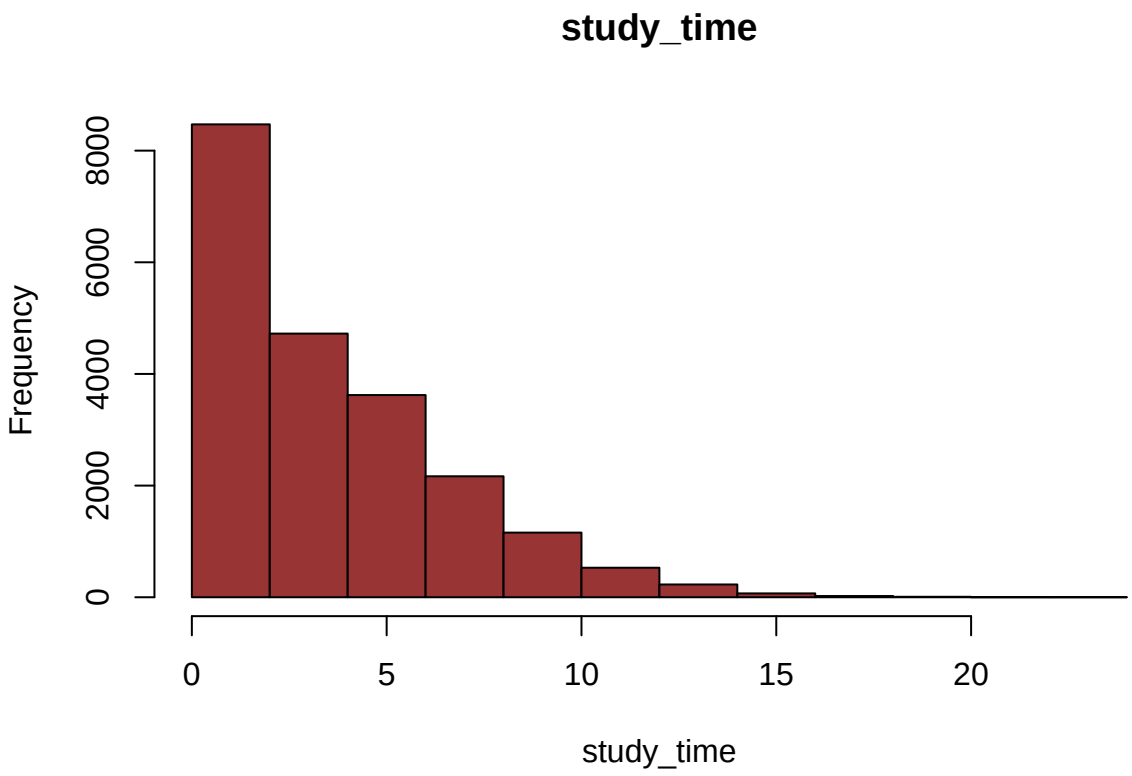
High school graduates are the most common.

```
# Histogram for service_length
hist(train_data_eng$service_length, main="service_length", xlab="service_length", col="#993435")
```

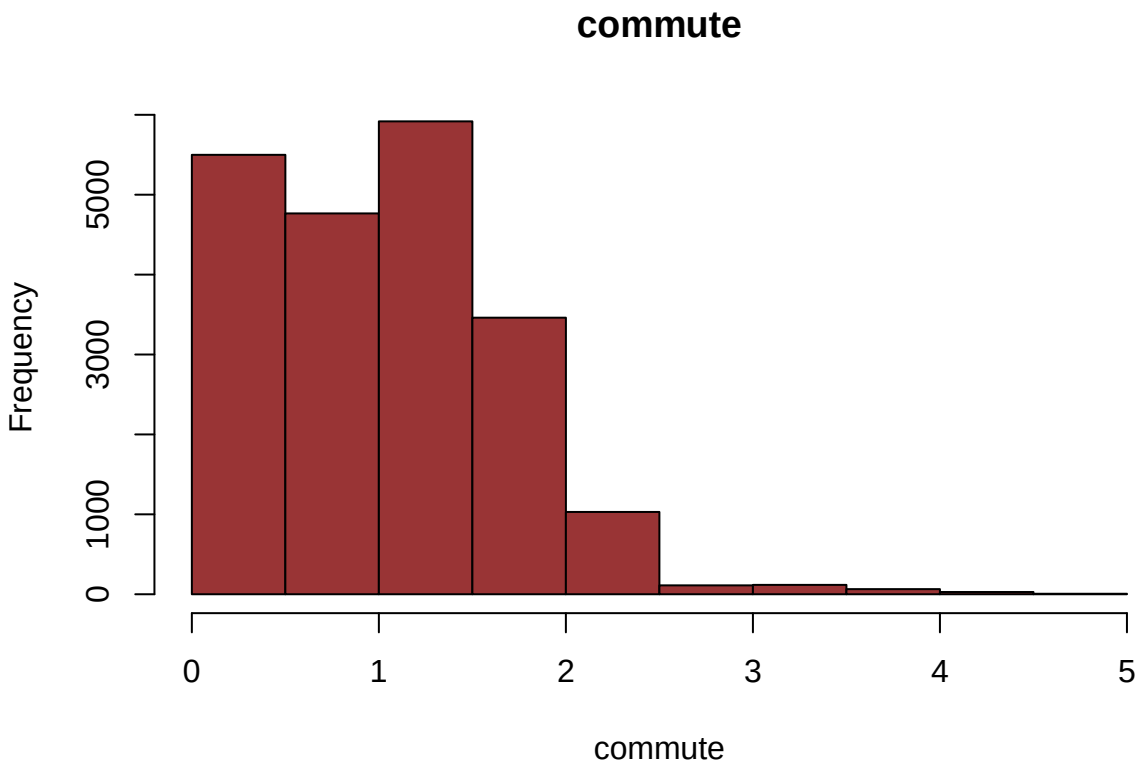


This column is expected to have a strong correlation with age.

```
# Histogram for study_time
hist(train_data_eng$study_time, main="study_time", xlab="study_time", col="#993435")
```

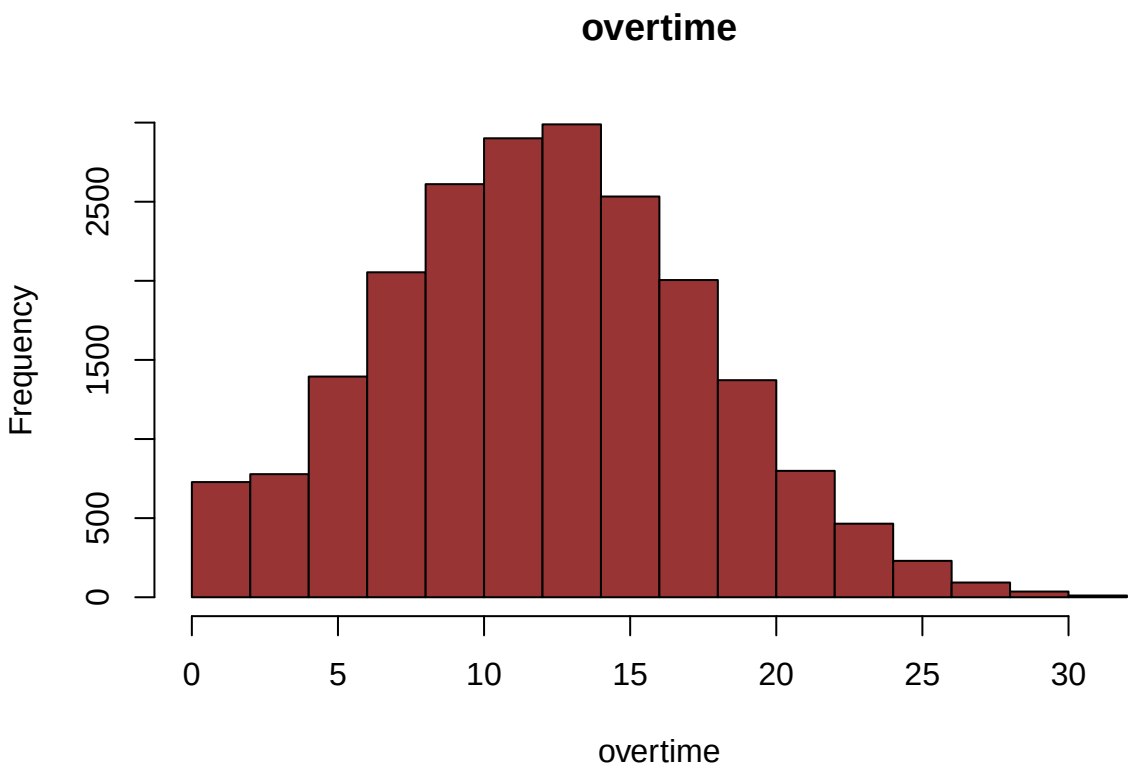



```
# Histogram for commute
hist(train_data_eng$commute, main="commute", xlab="commute", col="#993435")
```



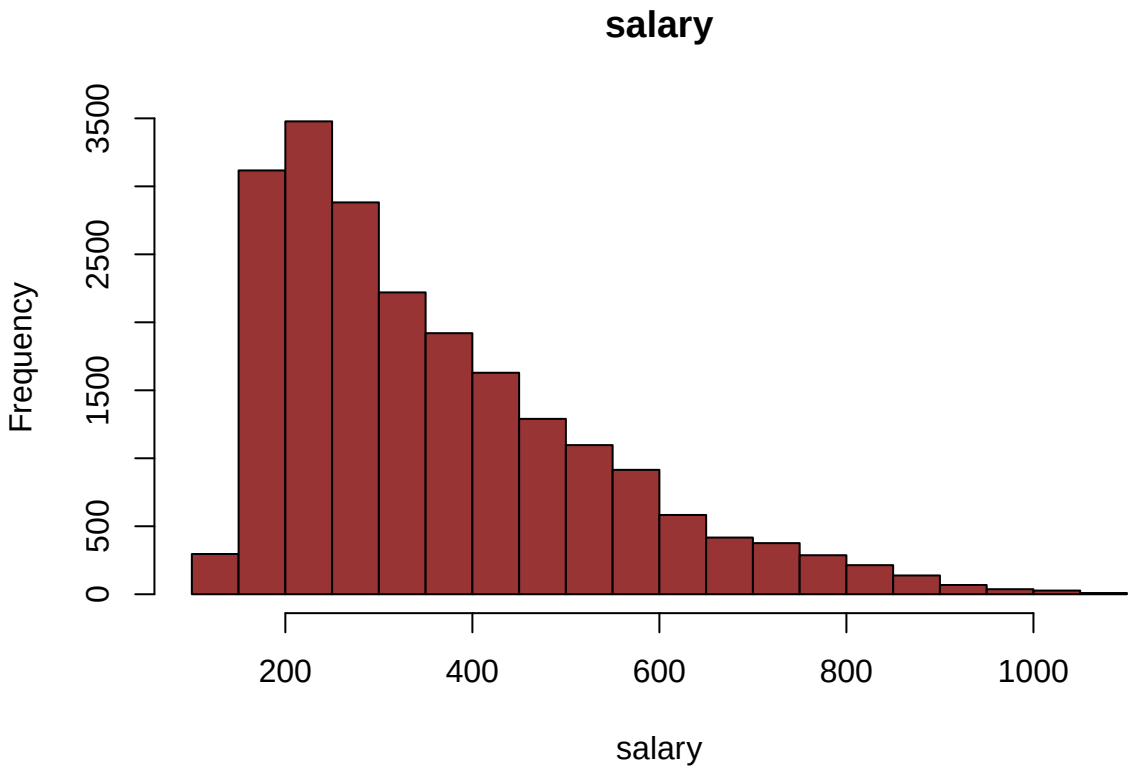
This column is expected to have a strong connection with the area.

```
# Histogram for overtime
hist(train_data_eng$overtime, main="overtime", xlab="overtime", col="#993435")
```



It seems to be related to position and age.

```
# Histogram for salary
hist(train_data_eng$salary, main="salary", xlab="salary", col="#993435")
```



This column, which is the objective variable, has a downward-sloping distribution.

Create correlation matrix

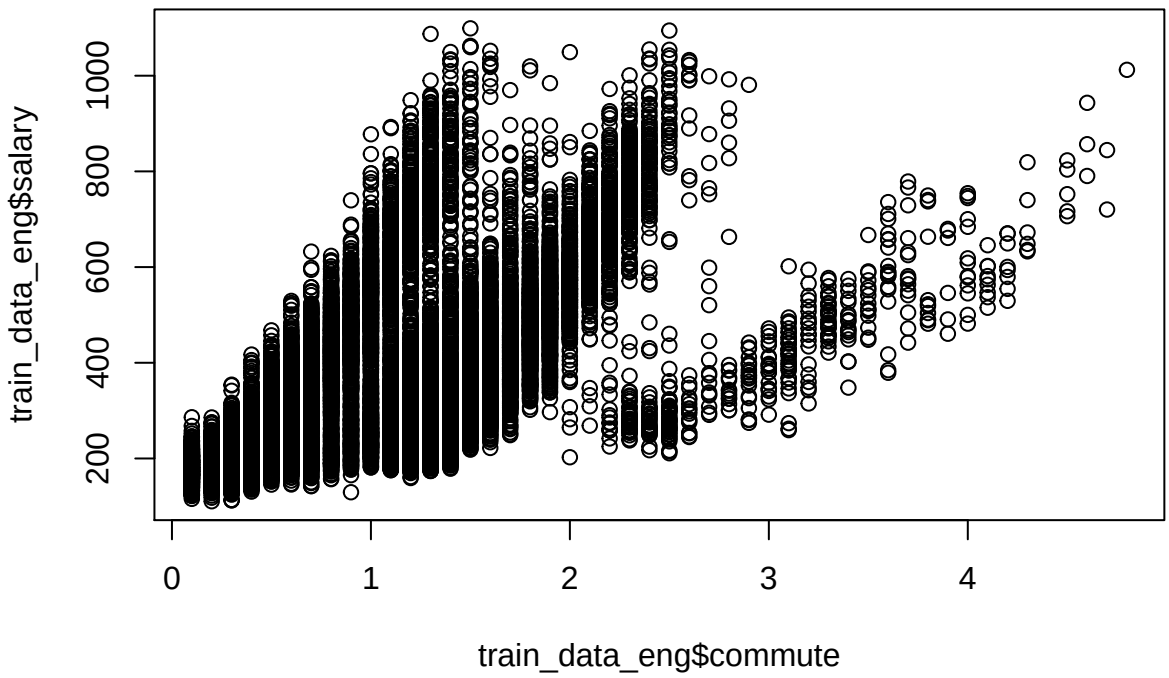
Note: Execute the following code to calculate the correlation matrix at once. However, due to the heavy processing, please run only high-performance PCs.

```
# Execute the following code to calculate the correlation matrix at once. However, due to the heavy pro

# psych::pairs.panels(train_data_eng[-1])
```

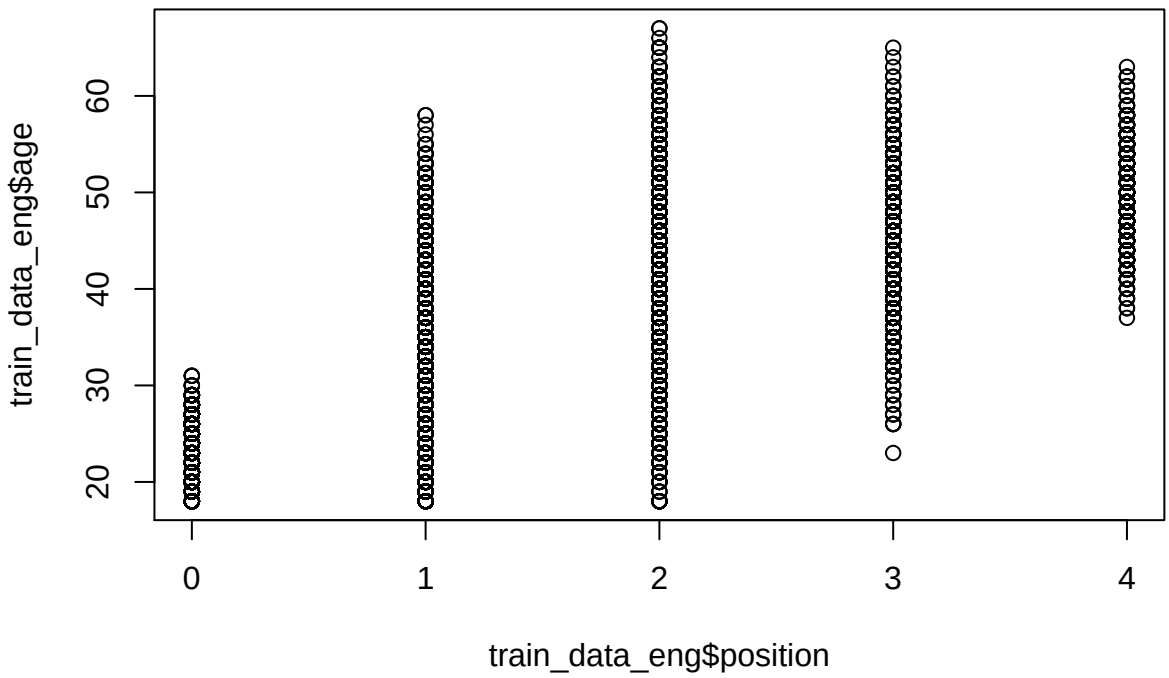
Check relationships between data using scatter plots

Check the distribution between two variables that are likely to have strong correlations or special relationships.
Note: This code is not necessary if the code of the previous correlation matrix was executed.

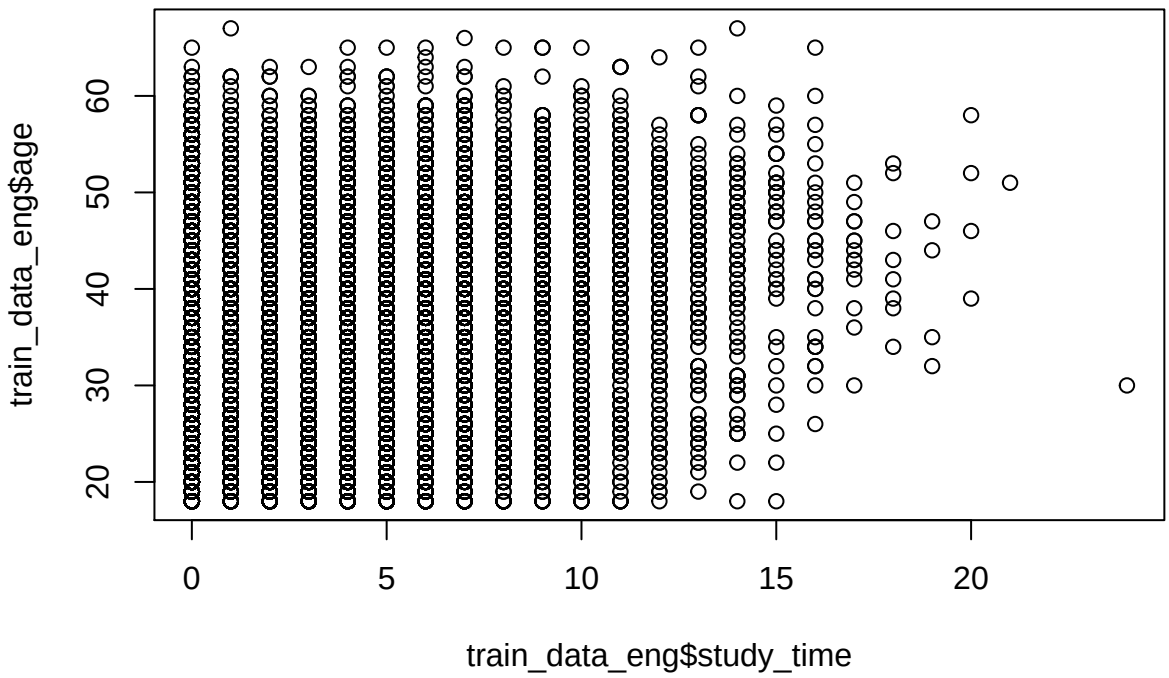
```
# Scatter plot of "commute" and "salary"
plot(train_data_eng$commute,train_data_eng$salary)
```



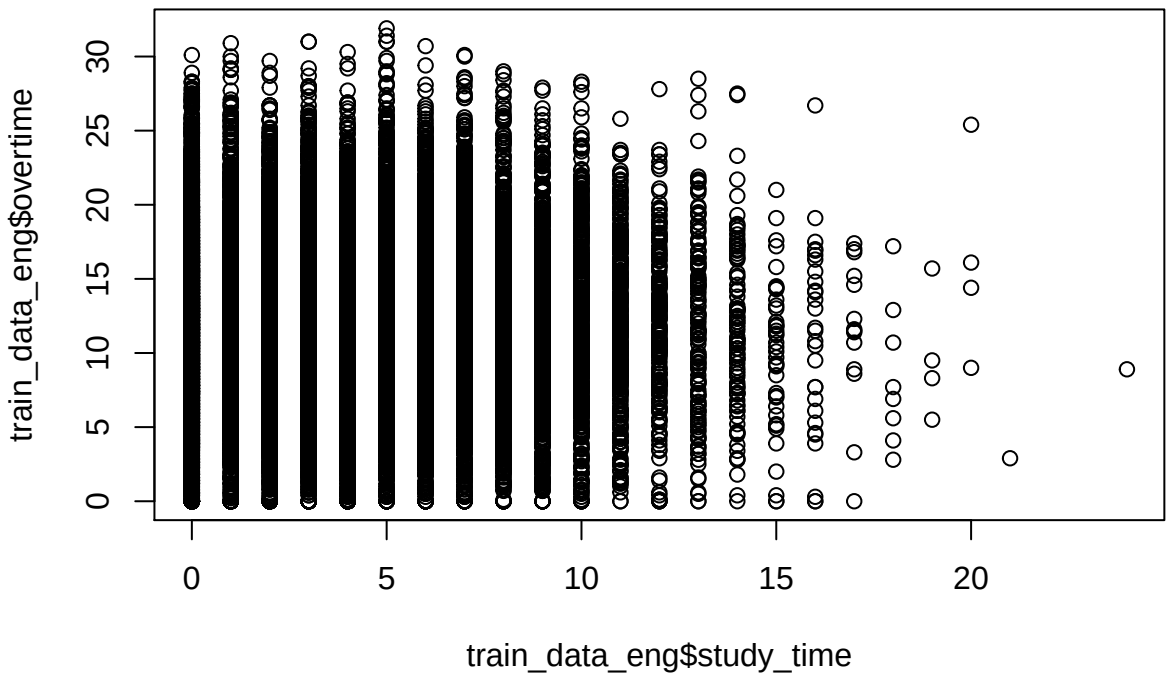

```
# Scatter plot of "position" and "age"
plot(train_data_eng$position,train_data_eng$age)
```



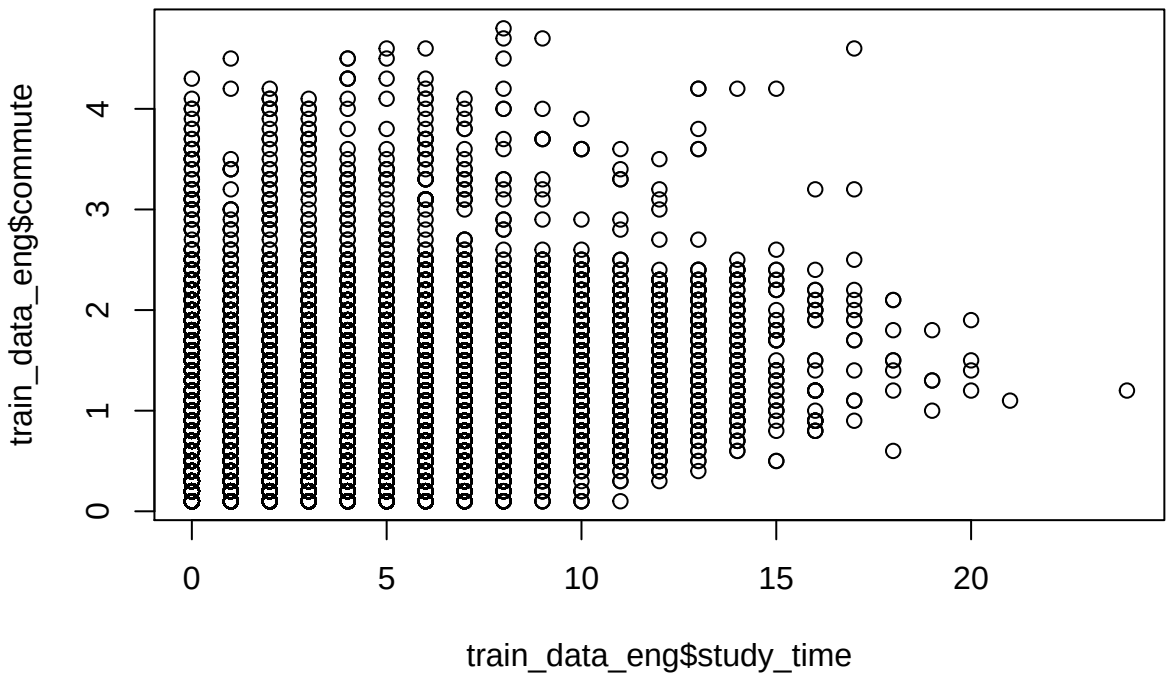

```
# Scatter plot of "study_time" and "age"
plot(train_data_eng$study_time,train_data_eng$age)
```



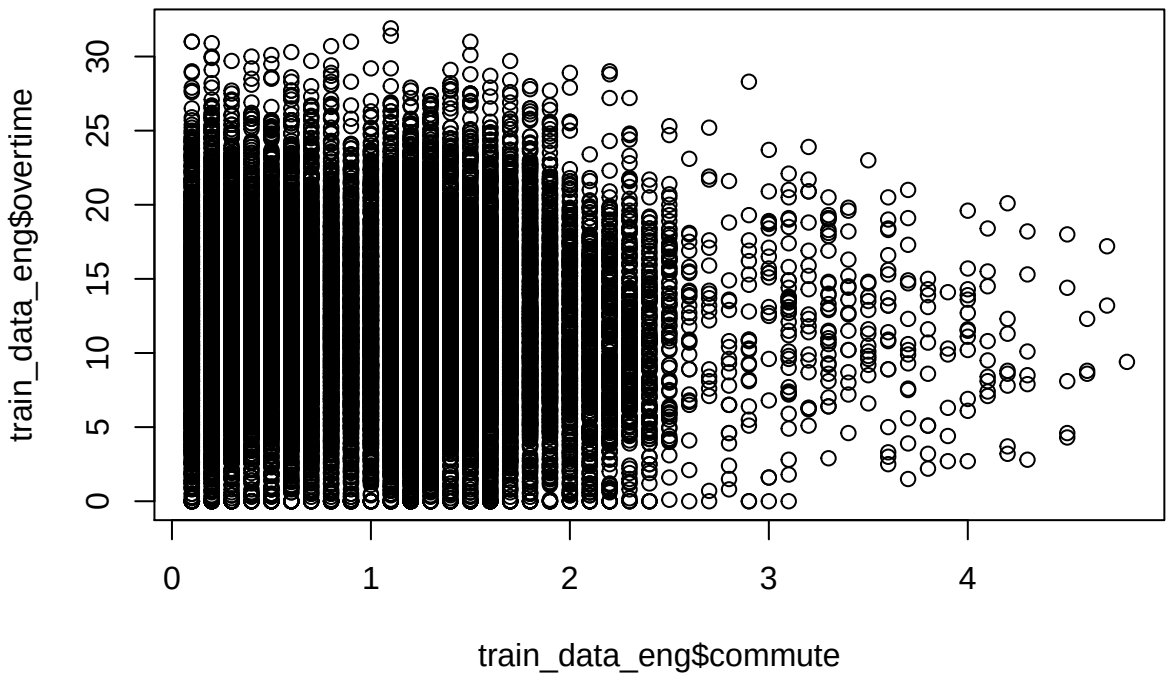

```
# Scatter plot of "study_time" and "overtime"
plot(train_data_eng$study_time,train_data_eng$overtime)
```




```
# Scatter plot of "study_time" and "commute"
plot(train_data_eng$study_time,train_data_eng$commute)
```



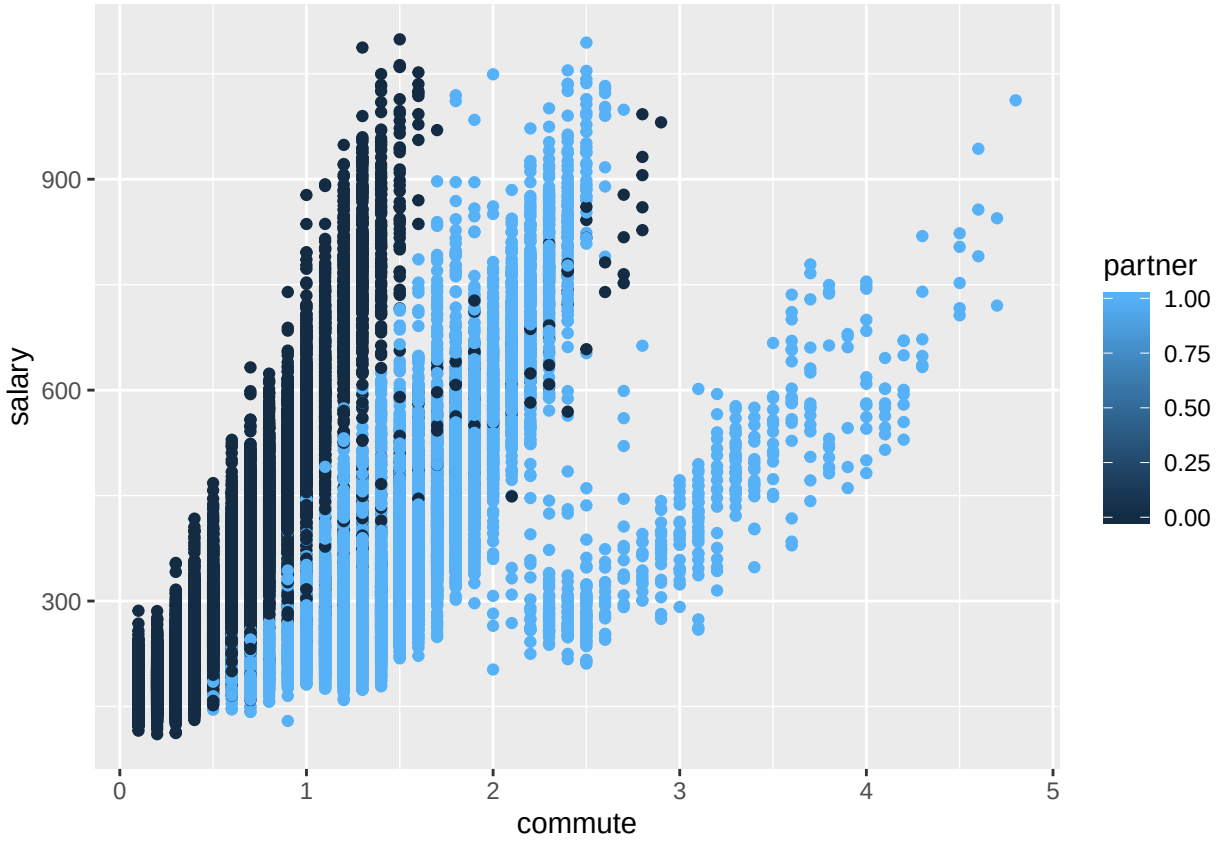

```
# Scatter plot of "commute" and "overtime"
plot(train_data_eng$commute,train_data_eng$overtime)
```



It seems that the higher the age, the higher the position.

Of particular note here are the scatter plots of “commute” and “salary”, which show a special tendency. Multiple groups appear to be represented in one scatter plot.

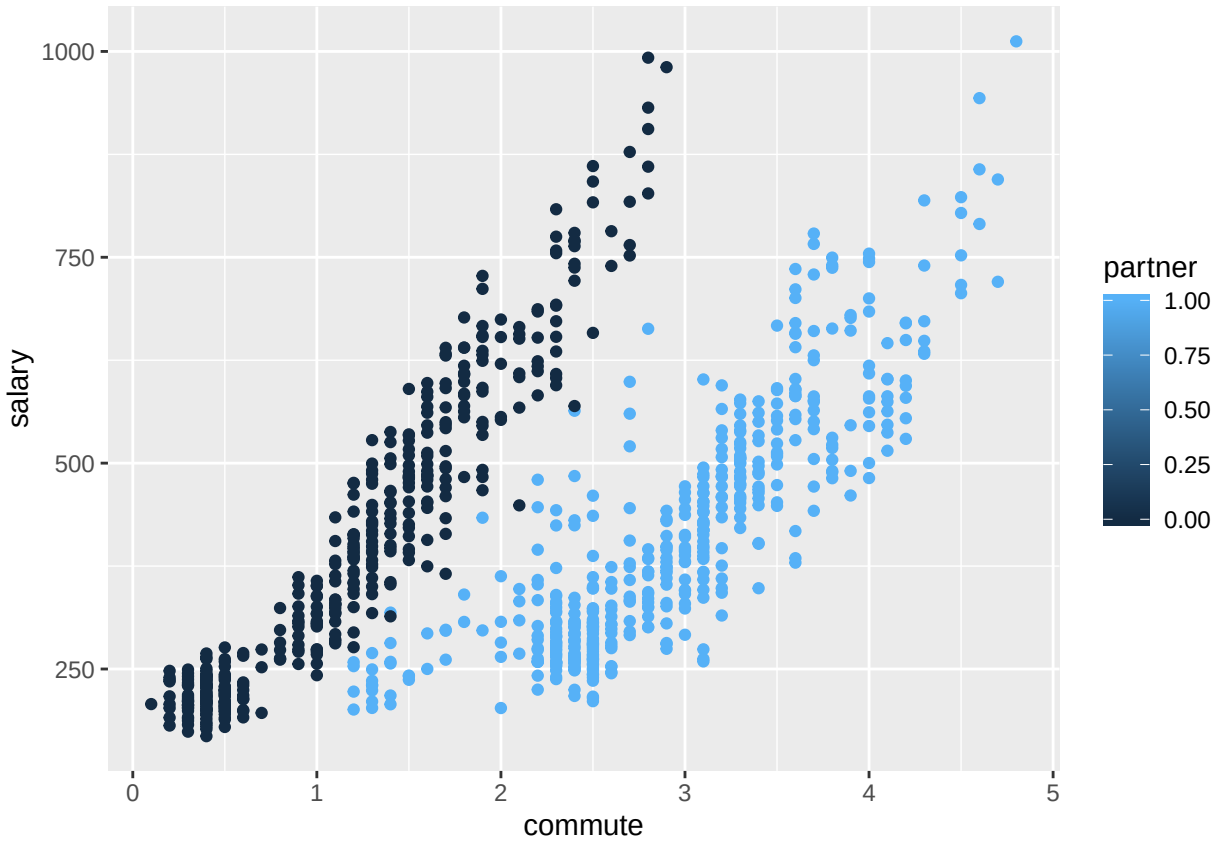
```
# Scatter plot of "commute" and "salary" color-coded by partner status
train_data_eng %>% ggplot(aes(y = salary, x = commute)) + geom_point(aes(colour=partner))
```



Partner status may explain some of the hidden relationships.

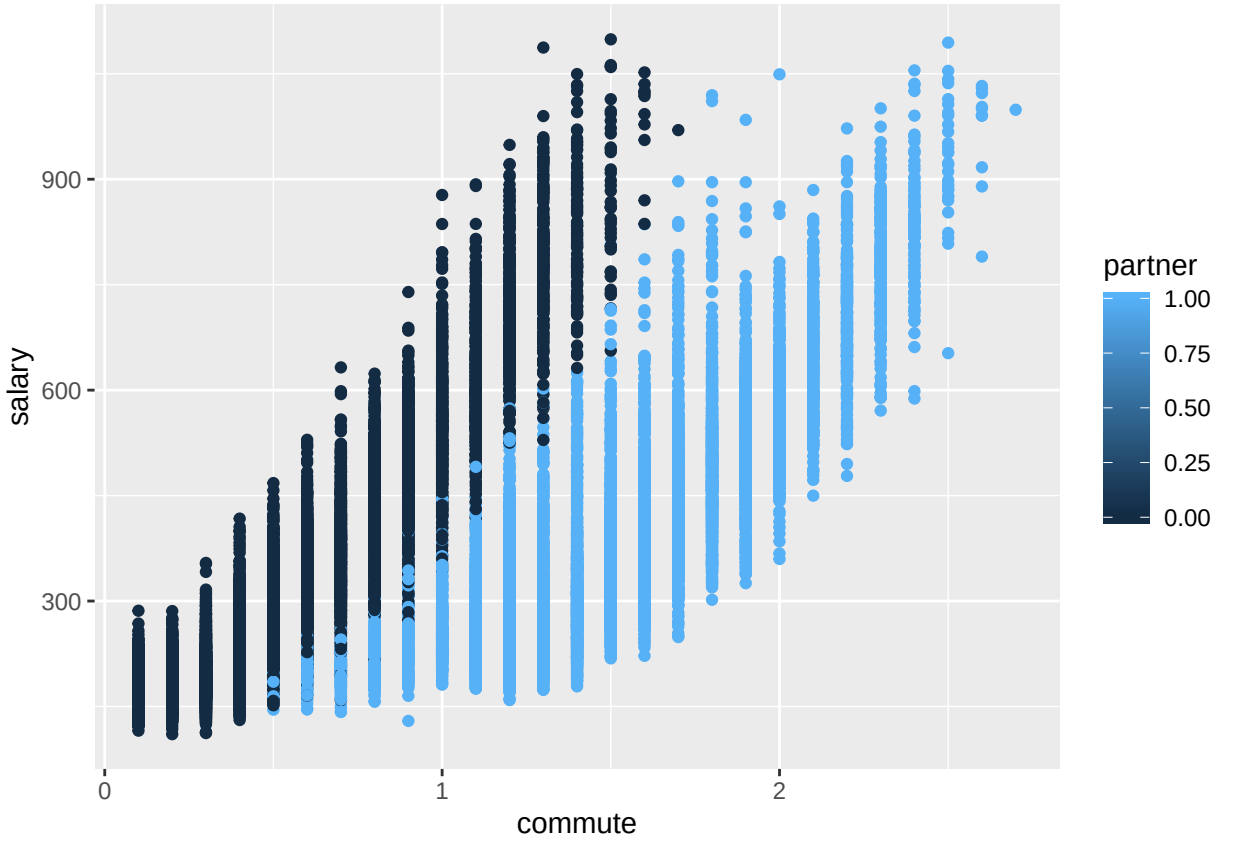
Next, try grouping by area. Area “13” and Area “27” are Tokyo and Osaka Prefecture. These are the two major cities that represent Japan.

```
# Scatter plot of "commute" and "salary" color-coded by partner status(urban resident only)
train_data_eng %>% filter(area == 13 | area == 27) %>% ggplot(aes(y = salary, x = commute)) + geom_poin
```



By area and partner status, the commute and salary scatter plots can be neatly split.

```
# Scatter plot of "commute" and "salary" color-coded by partner status(non-urban resident only)
train_data_eng %>% filter(area != 13 & area != 27) %>% ggplot(aes(y = salary, x = commute)) + geom_poin
```



Similar trends are seen in non-urban areas.

The data show that trends vary greatly depending on the existence of partners and whether they are urban residents. Therefore, a new column segmented by these two points is added.

Below, the description of the newly added column.

segment column meanings “1” for non-urban residents and no partner. “2” for non-urban residents and having a partner. “3” for urban residents and no partner. “4” for urban residents and having a partner.

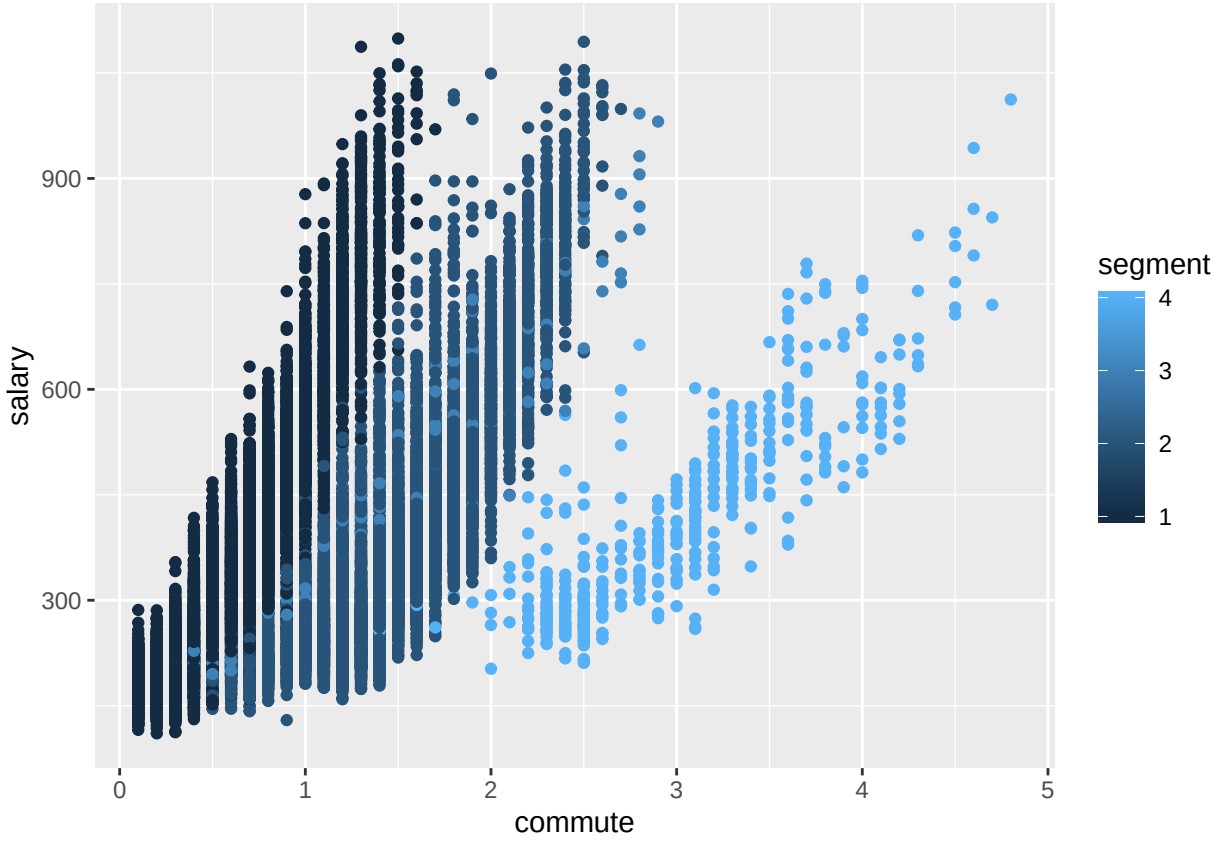
```
# Create a new dataset with additional columns called segments based on area and partner status
train_data_eng_segment <- train_data_eng %>% mutate(.,segment =
                                                    case_when(
                                                      area != 13 & area != 27 & partner == 0 ~ 1,
                                                      area != 13 & area != 27 & partner == 1 ~ 2,
                                                      (area == 13 | area == 27) & partner == 0 ~ 3,
                                                      (area == 13 | area == 27) & partner == 1 ~ 4
                                                    ))

# Check data
head(train_data_eng_segment)
```

```
## # A tibble: 6 x 14
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        1    44    23     2       1         2         1
## 2     1        2    31    29     1       0         0         0
## 3     2        2    36    35     1       0         0         2
## 4     3        0    22    13     2       0         0         0
```

```
## 5     4         0    25    46     2        0         0         1
## 6     5         1    23    28     2        1         3         1
## # ... with 6 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, salary <dbl>, segment <dbl>
```

Create train_data_eng_segment dataset with this column added.

```
# Scatter plot of "commute" and "salary" color-coded by added segment
train_data_eng_segment %>% ggplot(aes(y = salary, x = commute)) + geom_point(aes(colour=segment))
```



I want to verify that adding this column increases the accuracy of the model.

Add the same column to the test set and name it test_data_eng_segment.

```
# Create a new dataset with additional columns called segments in the test set
test_data_eng_segment <- test_data_eng %>% mutate(.,segment =
                                            case_when(
                                              area != 13 & area != 27 & partner == 0 ~ 1,
                                              area != 13 & area != 27 & partner == 1 ~ 2,
                                              (area == 13 | area == 27) & partner == 0 ~ 3,
                                              (area == 13 | area == 27) & partner == 1 ~ 4
                                            ))
```

Model Building

Simple mean model

Create a simple prediction model with the average salary value as the predicted value. Based on this, build a model while adding variables to increase accuracy.

```
# Calculate the average
mu <- mean(train_data_eng$salary)
# Create a sumit file that stores the average value
a <- data.frame(id = 0:8999, y = rep(mu, 9000))
# Export to CSV
write.csv(a, "submission_mean.csv", row.names=FALSE)
```

The MAE is 143.379 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- tibble(Model = "Model.1", Method="simple mean",
                                    MAE = 143.379)
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379

However, the distribution of salary is declining to the right, not a normal distribution. Since it is not appropriate to use the average value as the representative value in the case of non-normal distribution, a model using the median value as the predicted value is also created.

Simple median model

```
# Calculate the median
med <- median(train_data_eng$salary)
# Create a sumit file that stores the median value
b <- data.frame(id = 0:8999, y = rep(med, 9000))
# Export to CSV
write.csv(b, "submission_median.csv", row.names=FALSE)
```

The MAE is 134.864 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.2", Method="simple median",
                                   MAE = 134.864 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864

The MAE was lower in the model using the median than in the model using the mean.

position effect model

Next, create a model that incorporates the effects of position.

```
# Add position effect
pos_effects <- train_data_eng %>%
  group_by(position) %>%
  summarize(pos_effect = mean(salary - med))
# Check the effect of movie effect
pos_effects
```

```
## # A tibble: 5 x 2
##   position pos_effect
##      <int>      <dbl>
## 1        0      -93.8
## 2        1       36.8
## 3        2      147.
## 4        3      207.
## 5        4      321.
```

The results show that the higher the position, the higher the salary. Use this effect to predict salary.

```
# Predict salary
pred_salary <- test_data_eng %>%
  left_join(pos_effects, by='position') %>%
  mutate(pred = med + pos_effect)

# Check predicted values
head(pred_salary)
```

```
## # A tibble: 6 x 14
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        3    39    46     2       1         5         1
## 2     1        1    31     4     1       0         0         4
## 3     2        0    20    23     2       1         2         0
## 4     3        0    28    24     2       0         0         0
## 5     4        1    41    38     2       0         0         0
## 6     5        0    22    34     2       0         0         1
## # ... with 6 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, pos_effect <dbl>, pred <dbl>
```

```
# Create a sumit file that stores the predicted values
d <- data.frame(0:8999, pred_salary$pred)
colnames(d) <- c("id", "y")

# Export to CSV
write.csv(d, "submission_pos.csv", row.names=FALSE)
```

The MAE is 94.660 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.3", Method="position",
                              MAE = 94.660 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864

Model	Method	MAE
Model.3	position	94.660

position & sex effect model

In addition, the impact of salary on the sex will be incorporated into the model.

```
# Add sex effect
sex_effects <- train_data_eng %>%
  left_join(pos_effects, by='position') %>%
  group_by(sex) %>%
  summarize(sex_effect = mean(salary - med - pos_effect))
# Check the effect of movie effect
sex_effects
```

```
## # A tibble: 2 x 2
##     sex sex_effect
##   <int>      <dbl>
## 1     1       6.75
## 2     2      -6.80
```

The results show that men have slightly higher salary. Use this effect to predict salary.

```
# Predict salary
pred_salary <- test_data_eng %>%
  left_join(pos_effects, by='position') %>%
  left_join(sex_effects, by='sex') %>%
  mutate(pred = med + pos_effect + sex_effect)

# Check predicted values
head(pred_salary)
```

```
## # A tibble: 6 x 15
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        3    39    46     2       1         5         1
## 2     1        1    31     4     1       0         0         4
## 3     2        0    20    23     2       1         2         0
## 4     3        0    28    24     2       0         0         0
## 5     4        1    41    38     2       0         0         0
## 6     5        0    22    34     2       0         0         1
## # ... with 7 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, pos_effect <dbl>, sex_effect <dbl>,
## #   pred <dbl>
```

```
# Create a sumit file that stores the predicted values
d <- data.frame(0:8999, pred_salary$pred)
colnames(d) <- c("id", "y")

# Export to CSV
write.csv(d, "pos_sex.csv", row.names=FALSE)
```

The MAE is 84.660 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
```

```
                          tibble(Model = "Model.4", Method="pos & sex",
                                 MAE = 84.660 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864
Model.3	position	94.660
Model.4	pos & sex	84.660

position & sex & education effect model

Next, the effect of education level on salary is incorporated into the model.

```
# Add education effect
education_effects <- train_data_eng %>%
  left_join(pos_effects, by='position') %>%
  left_join(sex_effects, by='sex') %>%
  group_by(education) %>%
  summarize(education_effect = mean(salary - med - pos_effect - sex_effect))
# Check the effect of movie effect
education_effects
```

```
## # A tibble: 5 x 2
##   education education_effect
##       <int>            <dbl>
## 1         0            -34.0
## 2         1            -21.5
## 3         2             12.3
## 4         3             97.9
## 5         4            231.
```

The results show that the higher the educational background, the higher the salary. Especially the doctor is extremely high. Use this effect to predict salary.

```
# Predict salary
pred_salary <- test_data_eng %>%
  left_join(pos_effects, by='position') %>%
  left_join(sex_effects, by='sex') %>%
  left_join(education_effects, by='education') %>%
  mutate(pred = med + pos_effect + sex_effect + education_effect)

# Check predicted values
head(pred_salary)
```

```
## # A tibble: 6 x 16
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        3    39    46     2       1         5         1
## 2     1        1    31     4     1       0         0         4
## 3     2        0    20    23     2       1         2         0
## 4     3        0    28    24     2       0         0         0
## 5     4        1    41    38     2       0         0         0
## 6     5        0    22    34     2       0         0         1
```

```
## # ... with 8 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, pos_effect <dbl>, sex_effect <dbl>,
## #   education_effect <dbl>, pred <dbl>
```

```
# Create a sumit file that stores the predicted values
d <- data.frame(0:8999, pred_salary$pred)
colnames(d) <- c("id", "y")

# Export to CSV
write.csv(d, "pos_sex_edu.csv", row.names=FALSE)
```

The MAE is 74.660 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.5", Method="pos & sex & edu",
                              MAE = 74.660 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864
Model.3	position	94.660
Model.4	pos & sex	84.660
Model.5	pos & sex & edu	74.660

position & sex & education & segment effect model

Finally, create a model that incorporates the effect of the segment by area and partner status.

```
# Add education effect
segment_effects <- train_data_eng_segment %>%
  left_join(pos_effects, by='position') %>%
  left_join(sex_effects, by='sex') %>%
  left_join(education_effects, by='education') %>%
  group_by(segment) %>%
  summarize(segment_effect = mean(salary - med - pos_effect - sex_effect - education_effect))
# Check the effect of movie effect
segment_effects
```

```
## # A tibble: 4 x 2
##   segment segment_effect
##     <dbl>          <dbl>
## 1       1          -26.0
## 2       2           22.5
## 3       3           19.4
## 4       4           62.3
```

The results show that salary is higher in urban areas with partners. Singles in non-urban areas have lower salary. Use this effect to predict salary.

```
# Predict salary
pred_salary <- test_data_eng_segment %>%
  left_join(pos_effects, by='position') %>%
  left_join(sex_effects, by='sex') %>%
  left_join(education_effects, by='education') %>%
```

```
  left_join(segment_effects, by='segment') %>%
  mutate(pred = med + pos_effect + sex_effect + education_effect + segment_effect)

# Check predicted values
head(pred_salary)
```

```
## # A tibble: 6 x 18
##      id position   age  area   sex partner num_child education
##   <int>    <int> <int> <int> <int>   <int>     <int>     <int>
## 1     0        3    39    46     2       1         5         1
## 2     1        1    31     4     1       0         0         4
## 3     2        0    20    23     2       1         2         0
## 4     3        0    28    24     2       0         0         0
## 5     4        1    41    38     2       0         0         0
## 6     5        0    22    34     2       0         0         1
## # ... with 10 more variables: service_length <int>, study_time <int>,
## #   commute <dbl>, overtime <dbl>, segment <dbl>, pos_effect <dbl>,
## #   sex_effect <dbl>, education_effect <dbl>, segment_effect <dbl>,
## #   pred <dbl>
```

```
# Create a sumit file that stores the predicted values
d <- data.frame(0:8999, pred_salary$pred)
colnames(d) <- c("id", "y")

# Export to CSV
write.csv(d, "pos_sex_edu_seg.csv", row.names=FALSE)
```

The MAE is 64.660 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.6", Method="pos & sex & edu & seg",
                              MAE = 64.660 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864
Model.3	position	94.660
Model.4	pos & sex	84.660
Model.5	pos & sex & edu	74.660
Model.6	pos & sex & edu & seg	64.660

In the following, a model of machine learning is created using the caret package. What I want to try is a random forest, and I want to try three methods: “ranger”, “rf”, and “Rborist”.

ranger model

Here, the ranger method is used. At first I want to predict without using the created segment column.

```
if(0){
set.seed(123)
modelRanger <- train(
  salary ~  . -id, # not include ID in model
```

```
  data = train_data_eng, # There is no segment column
  method = "ranger", # Specifying the method to use
  tuneLength = 4, # Specify the range of parameter tuning
  preProcess = c('center', 'scale'), # Preprocessing to normalize data
  trControl = trainControl(method = "cv") # Use cross validation
)

# Create a sumit file that stores the predicted values
predRanger <- predict(modelRanger, test_data_eng)
d <- data.frame(0:8999, predRanger)
colnames(d) <- c("id", "y")
# Export to CSV
write.csv(d, "submission6.csv", row.names=FALSE)
}
```

The MAE is 24.660 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.7", Method="ranger",
                              MAE = 24.660 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864
Model.3	position	94.660
Model.4	pos & sex	84.660
Model.5	pos & sex & edu	74.660
Model.6	pos & sex & edu & seg	64.660
Model.7	ranger	24.660

ranger with segment model

Next, use the ranger method using the segment column.

```
if(0){
set.seed(123)
modelRanger <- train(
  salary ~  . -id, # not include ID in model
  data = train_data_eng_segment, # Use new dataset with added segment columns
  method = "ranger", # Specifying the method to use
  tuneLength = 4, # Specify the range of parameter tuning
  preProcess = c('center', 'scale'), # Preprocessing to normalize data
  trControl = trainControl(method = "cv") # Use cross validation
)
# Create a sumit file that stores the predicted values
predRanger <- predict(modelRanger, test_data_eng_segment)
d <- data.frame(0:8999, predRanger)
colnames(d) <- c("id", "y")
# Export to CSV
write.csv(d, "submission5.csv", row.names=FALSE)
}
```

The MAE is 23.614 as a result of submitting this file on the competition site.

```
# Store and display results in a list
mae_table <- bind_rows(mae_table,
                       tibble(Model = "Model.8", Method="ranger with segment",
                              MAE = 23.614 ))
mae_table %>% knitr::kable()
```

Model	Method	MAE
Model.1	simple mean	143.379
Model.2	simple median	134.864
Model.3	position	94.660
Model.4	pos & sex	84.660
Model.5	pos & sex & edu	74.660
Model.6	pos & sex & edu & seg	64.660
Model.7	ranger	24.660
Model.8	ranger with segment	23.614

The model using the segment column had higher accuracy.

The following two models ("rf", and "Rborist") could not be executed, but are described for reference.

rf model

Note: It is in the same state as commented out by enclosing it in an if statement so that it will not be executed.

```
# Use an if statement to put a comment out state
if(0){
set.seed(123)
modelRF <- train(
  salary ~ . -id, # not include ID in model
  data = train_data_eng,
  method = "rf", # Specifying the method to use
  tuneLength = 4, # Specify the range of parameter tuning
  preProcess = c('center', 'scale'), # Preprocessing to normalize data
  trControl = trainControl(method = "cv") # Use cross validation
)
# Create a sumit file that stores the predicted values
predRF <- predict(modelRF, test_data_eng)
d <- data.frame(0:8999, predRF)
colnames(d) <- c("id", "y")
# Export to CSV
write.csv(d, "submission8.csv", row.names=FALSE)
}
```

Rborist model

Note: It is in the same state as commented out by enclosing it in an if statement so that it will not be executed.

```
# Use an if statement to put a comment out state
if(0){
set.seed(123)
```

```r
modelRborist <- train(
  salary ~ . -id, # not include ID in model
  data = train_data_eng,
  method = "Rborist", # Specifying the method to use
  tuneLength = 4, # Specify the range of parameter tuning
  preProcess = c('center', 'scale'), # Preprocessing to normalize data
  trControl = trainControl(method = "cv") # Use cross validation
)
# Create a sumit file that stores the predicted values
predRborist <- predict(modelRborist, test_data_eng)
d <- data.frame(0:8999, predRborist)
colnames(d) <- c("id", "y")
# Export to CSV
write.csv(d, "submission9.csv", row.names=FALSE)
}
```

In this data format, ranger seems to be the fastest in random forest.

conclusion

It was found that accuracy was improved by performing feature engineering and creating meaningful variables in addition to existing variables. On the other hand, the use of the caret package requires a large computational load on a home PC, and it took a long time to complete the processing. It may be necessary to reduce or aggregate variables. This time, the submission of the competition was unlimited, so the prepared training data was not validated. When creating models in actual work or participating in competitions with submit restrictions, we should always create validations and verify accuracy. The memory problem may be solved by dividing the data set or selecting variables, but that is left for future work.

Participating in the competition with the model with the highest accuracy, ranked 78th out of 312 teams. Since it's a competition site for Python, we can't see the solution of other users using R, but in the forum we can see what models were created by the top rankers. Most rankers seemed to use LightGBM or NN. In particular, there appeared to be differences in the accuracy of feature engineering, and the ingenuity was different for each user.