R

[R] R 을 활용한 데이터 탐색(Exploratory Data Analysis)

2020. 7. 30. 00:02 수정 삭제 공개

EDA(Exploratory Data Analysis)

수집한 데이터가 들어왔을 때, 이를 다양한 각도에서 관찰하고 이해하는 과정입니다. 한마디로 데이터를 분석하기 전에 그래프나 통계적인 방법으로 자료를 직관적으로 바라보는 과정입니다.

여기서는 7가지 단계로 EDA로 하겠습니다.

1.데이터 탐색

2.결측치(NA) 처리

3.이상치(outlier) 발견 처리

4.리코딩(코딩 변경)

5.파생변수, 가변수

6.시각화

7.의사결정

※여기서 의사결정 부분은 데이터를 통해 결정하는 과정으로 생략하겠습니다.

1.데이터 탐색

데이터를 확인하는 작업

```
107  str(dataset)
108  head(dataset)          #6개만 출력
109  dim(dataset)           #행과 열의 개수
110  nrow(dataset)          #행의 개수
111  ncol(dataset)          #속성의 개수
112  length(dataset)        #속성의 길이
113  names(dataset)         #컬럼값
114  summary(dataset$price) #요약
```

위와 같은 명령어로 데이터를 확인한다.

2.결측치(NA) 처리

데이터 분석을 하는데 방해가 되는 NA 값을 처리한다.
결측치는 보통 중위수나 평균값 또는 0 / 삭제 이런 방식을 사용한다.

```
> #결측치 처리하는 방법
> # 중위수나 평균값으로 / 0으로 / 삭제
> # caret: na.omit() - 전체의 데이터셋을 대상으로 결측치를 처리한다.
> dataset_new ← na.omit(dataset)
> table(is.na(dataset_new))

FALSE
 1463
```

```r
#결측치 처리하는 방법
# 중위수나 평균값으로 / 0으로 / 삭제

# caret: na.omit() - 전체의 데이터셋을 대상으로 결측치를 처리한다.
dataset_new <- na.omit(dataset)
table(is.na(dataset_new))
str(dataset_new)

# 평균
mean(dataset$price, na.rm = T)

# 0으로 대체
price <- ifelse(is.na(dataset$price), 0, dataset$price)

# 평균으로 대체
price <- ifelse(is.na(dataset$price), mean(dataset$price, na.rm = T), dataset$price)

# 통계적 방법
```

예제)

```r
# type : 1, 2, 3
# TYPE이 1일 때는 price 값을 15%/10%/5% 이렇게 올린다.

# 300 고객유형을 판단하여 새로운 가변수 pricesta 만들어 처리된 값을 넣어보자
# type에 따라 평균값을 넣어주는 구문 / 평균값을 구하는데 있어서 결측치는 제외하고 넣는다.

for (i in 1:nrow(dataset_new)){
  if(dataset_new$type[i] == 1){
    dataset_new$pricesta[i] <-  mean(dataset$price, na.rm = T) * 1.15
  }else if(dataset_new$type[i] == 2){
    dataset_new$pricesta[i] <- mean(dataset$price, na.rm = T) * 1.10
  }else{
    dataset_new$pricesta[i] <- mean(dataset$price, na.rm = T) * 1.05
  }
}

#type 에 따라 price 의 값을 넣어주는 구문 / 여기서 결측처리를 해서 값이 없으면 평균을 넣고 있을 경우에는 price를 넣는다.
for (i in 1:nrow(dataset_new)){
  if(dataset_new$type[i] == 1){
    dataset_new$pricesta[i] <-  (ifelse(is.na(dataset_new$price[i]), mean(dataset_new$price[i], na.rm = T), dataset_new$price[i])) * 1.15
  }else if(dataset_new$type[i] == 2){
    dataset_new$pricesta[i] <- ifelse(is.na(dataset_new$price[i]), mean(dataset_new$price[i], na.rm = T), dataset_new$price[i]) * 1.10
  }else{
    dataset_new$pricesta[i] <- ifelse(is.na(dataset_new$price[i]), mean(dataset_new$price[i], na.rm = T), dataset_new$price[i]) * 1.05
  }
}

dataset_new
```

```r
#예제
# age 체크
summary(dataset$age)

# age 결측치 제거 후 boxplot으로 시각화

subset(dataset$age, na.rm=T)

# 결측치 날리는 방법
dataset_new2 <- na.omit(dataset$age)
dataset_new3 <- dataset$age[!is.na(dataset$age)]

summary(dataset_new2)

boxplot(dataset_new2,horizontal= T)
```

3.이상치(outlier) 발견 처리

결측치를 처리한 후에는 이상치를 발견 후 처리해야된다. 이 데이터들은 분석을 하는데 있어서 악영향을 미친다.

1)변수 유형이 이산변수인경우

```
# 3.이상치(outlier) 발견 처리

gender ← dataset$gender
range(gender)
table(gender)
```

gender라는 성별의 속성을 확인하고 있다.

```
> gender
  [1] 1 1 1 2 1 1 2 1 1 1 2 1 1 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1 2 2 2 2 2 1 2 2 2 1 2 1 2
 [77] 2 2 2 1 1 2 2 2 1 1 1 2 1 2 2 2 2 2 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 2 1 2 1 2 2 2 1 2 2 1 2 2 2 1 2 2 2 1 2 2 2 1 1 2 1 2 1 1 1 1 1 1 2 1 1 1 2 1 1 1 1 2
[153] 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 1 1 2 1 1 1 2 1 2 2 2 1 1 2 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 2 1 1 1 1 1 2 1 1 2 1 2 1 2 2 2 2 1
[229] 2 2 2 1 2 2 2 2 1 1 2 2 1 2 2 2 2 1 2 1 1 2 2 2 2 2 1 2 1 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 1 1 2 2 2 2 1 2 1 2 1 2 2 2 2 0 2 1 5 1 1 0 1
> range(gender)
[1] 0 5
> table(gender)
gender
  0   1   2   5
  2 173 124   1
```

gender는 범주형 데이터로서 형태를 봤을때 1,2 가 아닌 이상치인 0과 5가 들어있다 이걸 지워줘야한다.

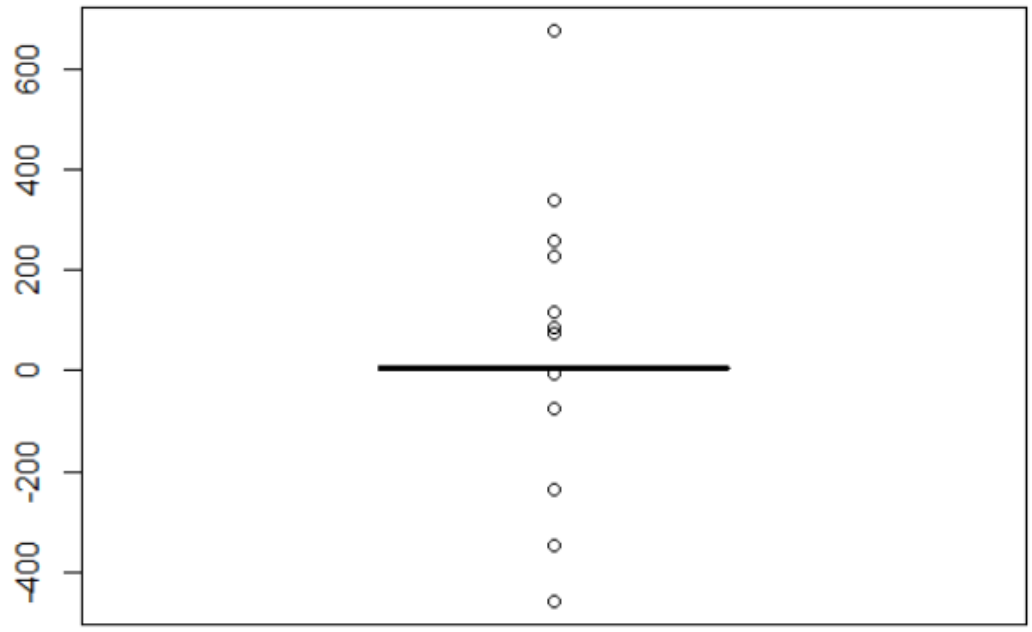
```
# subset을 활용하여 이상치를 제거한 후 gender를 범주형으로 변환해보자
dataset ← subset(dataset,gender == 1|gender == 2) # subset을 통해 gender가 1 or 2인 값만 가져와서 다시 dataset에 넣어준다.
dataset$gender ← factor(dataset$gender)           #factor 함수를 통해 타입을 factor형으로 바꾼다.
levels(dataset$gender)                            #levels 함수를 통해 해당 컬럼의 레벨이 어떻게 되어있는지 확인한다.
table(dataset$gender)                             #table 함수로 값이 어떻게 들어있는지 확인한다.
pie(table(dataset$gender))                        #pie 차트로 확인한다.
dim(dataset)                                      # 행과 열의 개수를 확인한다
str(dataset)
```

위에서는 1과2인 값을 추출해서 다시 넣어주는 방식을 통해 이상치를 제거했다.

2)변수 유형이 연속 변수 인 경우

```
> # 변수의 유형이 연속변수인 이상치 제거
>
> seqPrice ← dataset$price
> length(seqPrice)
[1] 297
> summary(seqPrice)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
-457.200    4.400    5.400    8.784    6.300  675.000       30
> #IQR (Q3-Q1) 6.2 - 4.6 = 1.6
> boxplot(seqPrice) # 시각화 해서 이상치 값을 확인
>
```

위의 속성을 summary를 했을때 사분위를 확인 했을때 이상치가 들어있는 것을 볼 수 있다.



boxplot으로 시각화해서 봤을때 이상치가 들어있는 것을 확인 할 수 있다.

```
> outlier ← boxplot(seqPrice)
> outlier  # 리스트 형식으로 출력된다.
$stats
     [,1]
[1,]  2.1
[2,]  4.4
[3,]  5.4
[4,]  6.3
[5,]  7.9

$n
[1] 267

$conf
         [,1]
[1,] 5.216281
[2,] 5.583719

$out
 [1]  675.0  257.8  -75.0   85.1 -457.2   -5.9  115.7   75.1  336.5  225.8   -4.8   85.1 -345.6   -5.9  115.7   75.1 -235.8 -235.8  336.5

$group
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

$names
[1] "1"
```

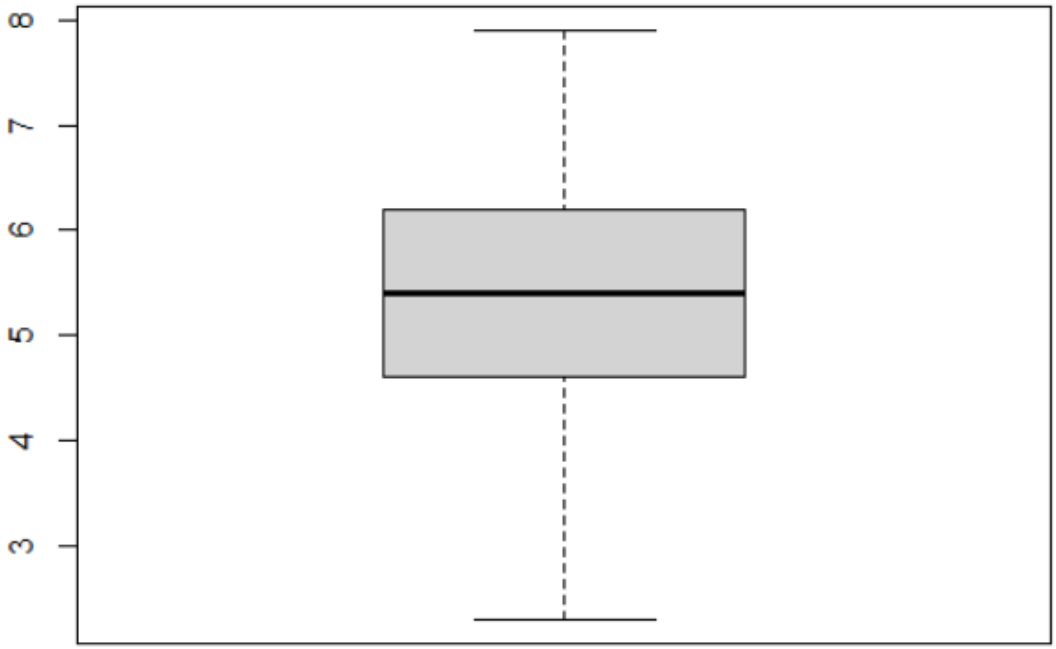
boxplot을 변수에 담고 출력 했을 때 위와 같이 나온다.

outlier를 봤을때 일반적인 값을 status의 범위에 있는 값들을 기준으로 범위를 정한다. 이상치를 제거 한후 다시 boxplot을 하면

```
> dataset ← subset(dataset , seqPrice ≥2.3 & seqPrice≤ 7.9)
> nrow(dataset)
[1] 247
```

status 범위에 있는 데이터만 추출하는 작업을 가진다.

```
> boxplot(dataset$price)
> |
```



위와같이 이상치가 제거된 걸 볼 수있다.

4.데이터 리코딩 / 파생변수, 가변수

아래에 있는 resident라는 속성값은 연속형으로 나타나 있지만 가독성을 위해서 범주형 데이터로 바꿔주었다.
그리고 이 데이터를 새로운 파생변수 resident_new라는 곳에 넣어주었다.

```r
# 리코딩 - 데이터의 가독성을 위해서
# 연속형 → 범주형
# 형식) dataset$컬럼[조건식] ← 추가할 값
# 1 : 서울 , 2: 부산, 3: 광주 , 4: 대전 , 5 : 대구
head(dataset$resident)
summary(dataset$resident)

dataset$resident[is.na(dataset$resident)]

dataset$resident_new[dataset$resident==1] <- "서울"
dataset$resident_new[dataset$resident==2] <- "부산"
dataset$resident_new[dataset$resident==3] <- "광주"
dataset$resident_new[dataset$resident==4] <- "대전"
dataset$resident_new[dataset$resident==5] <- "대구"
dataset$resident_new

#주거지의 NA 값을 행정수도인 대전으로 대체

dataset$resident_new[is.na(dataset$resident)] <- "대전"
dataset$resident_new

dataset$resident_new <- factor(dataset$resident_new)
levels(dataset$resident_new)
str(dataset$resident_new)
```

#--간단한 예제 1

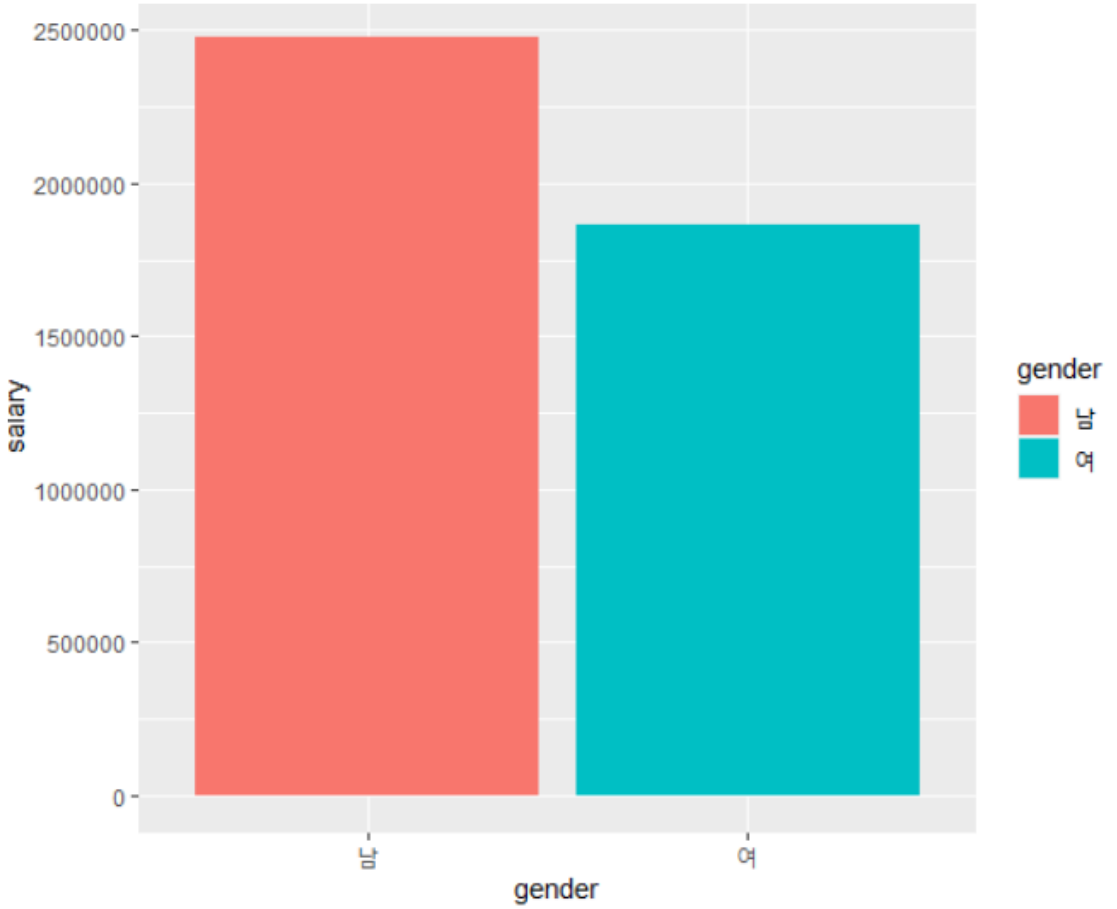
```r
url <-"https://www.dropbox.com/s/0djexymb42zd1e2/example_salary.csv?dl=1"
salary_data_eda <- read.csv(url, stringsAsFactors=F ,na="-") # - 이 값은 na로 취급한다.

# 1.컬럼명을 영문으로 변경
names(salary_data_eda) <- c("age", "Monthly salary" , "Annual special salary amount" , "Working hours","The number of workers","Career classification","ge
names(salary_data_eda)

# 2.각 피처별 결측값 확인
is.na(salary_data_eda$age)
is.na(salary_data_eda$`Monthly salary`)
is.na(salary_data_eda$`Annual special salary amount`)
is.na(salary_data_eda$`Working hours`)
is.na(salary_data_eda$`The number of workers`)
is.na(salary_data_eda$`Career classification`)
is.na(salary_data_eda$`gender`)

salary_data_eda$age[is.na(salary_data_eda$age)]
salary_data_eda$`Monthly salary`[is.na(salary_data_eda$`Monthly salary`)]
salary_data_eda$`Annual special salary amount`[is.na(salary_data_eda$`Annual special salary amount`)]
salary_data_eda$`Working hours`[is.na(salary_data_eda$`Working hours`)]
salary_data_eda$`The number of workers`[is.na(salary_data_eda$`The number of workers`)]
salary_data_eda$`Career classification`[is.na(salary_data_eda$`Career classification`)]
salary_data_eda$`gender`[is.na(salary_data_eda$`gender`)]


# 3.임금 평균 확인
salary_mean <- mean(salary_data_eda$`Monthly salary`,na.rm="T")
salary_mean
```

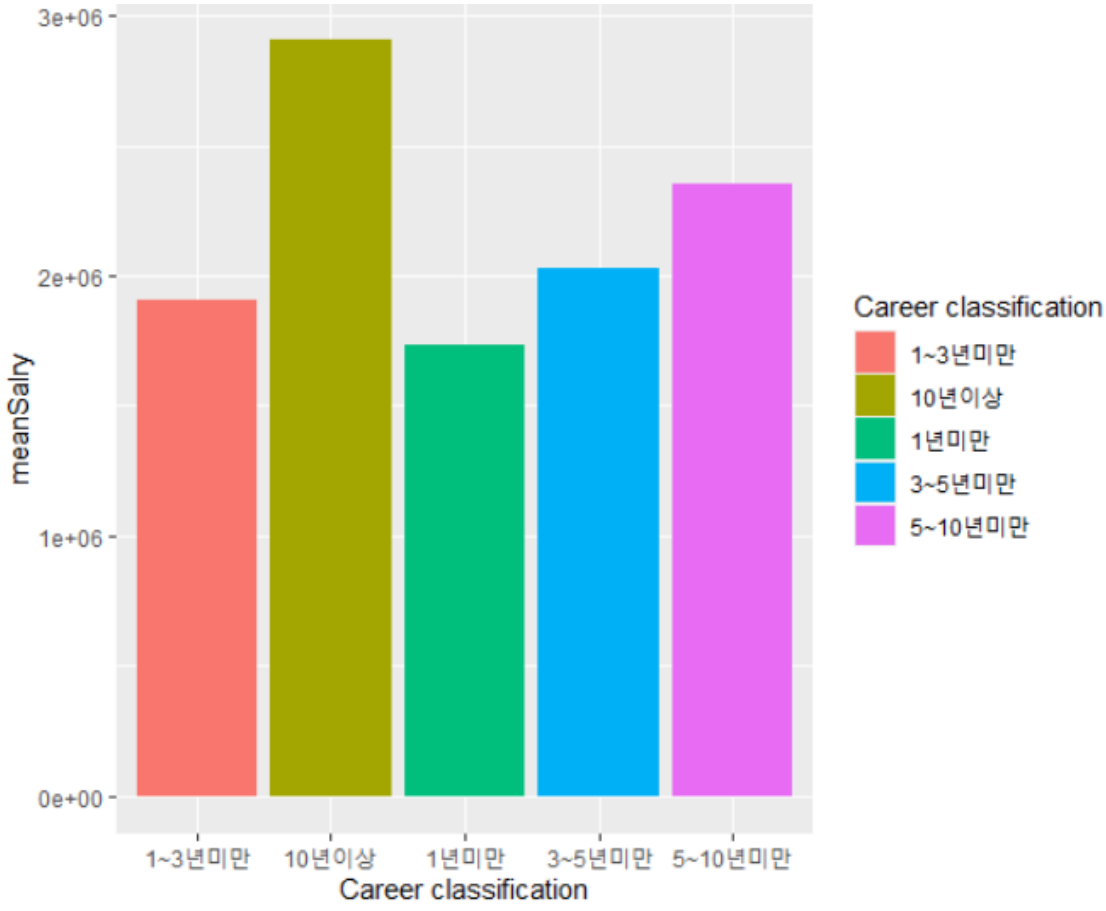
```r
# 4.임금 중앙값 확인
summary(salary_data_eda$`Monthly salary`)
salary_median<-median(salary_data_eda$`Monthly salary`,na.rm=T)
salary_median

# 5.임금 범위 구해보기(최저,최고)

range(salary_data_eda$`Monthly salary`,na.rm=T)

#범위에 들어있는 값 구하기
salaryRange <- subset(salary_data_eda$`Monthly salary`,
                      salary_data_eda$`Monthly salary` >=range(salary_data_eda$`Monthly salary`,na.rm=T)[1]
                      & salary_data_eda$`Monthly salary` <=range(salary_data_eda$`Monthly salary`,na.rm=T)[2])

salaryRange

#최소 최대 범위를 구하는 range
range(salary_data_eda$`Monthly salary`,na.rm=T)[1]

# 6.임금에 대한 사분위수(quantile())
quantile(salaryRange)

# 7.성별에 따른 임금 격차 확인해보기
genderGapSalary<-salary_data_eda %>%
  select(gender,`Monthly salary`)%>%
  group_by(gender) %>%
  summarise(salary = mean(`Monthly salary`,na.rm=T))

genderGapSalary
```

```r
# 8.분석된 데이터를 가지고 원하는 시각화 진행
ggplot(genderGap,aes(x=gender,y=salary,fill=gender))+
  geom_col()

# 9.성별에 따른 표준편차구하기
genderGapSd<-salary_data_eda %>%
  select(gender,`Monthly salary`)%>%
  group_by(gender) %>%
  summarise(sd = sd(`Monthly salary`,na.rm=T))
genderGapSd


# 10.경력별 임금 평균치
CareerMeanSalry <- na.omit(salary_data_eda) %>%
  select(`Monthly salary`,`Career classification`) %>%
  group_by(`Career classification`) %>%
  summarise(meanSalry = mean(`Monthly salary`))
CareerMeanSalry


# 11.경력별 임금 평균치 시각화
ggplot(CareerMeanSalry, aes(x=`Career classification`, y=meanSalry, fill=`Career classification` ))+
  geom_col()
```

6.시각화

이렇게 구한 데이터를 원하는 형태의 차트나 도표로 시각화해서 보여준다.


2500000
2000000
1500000
1000000
500000
0
salary
남
여
gender
gender
남
여

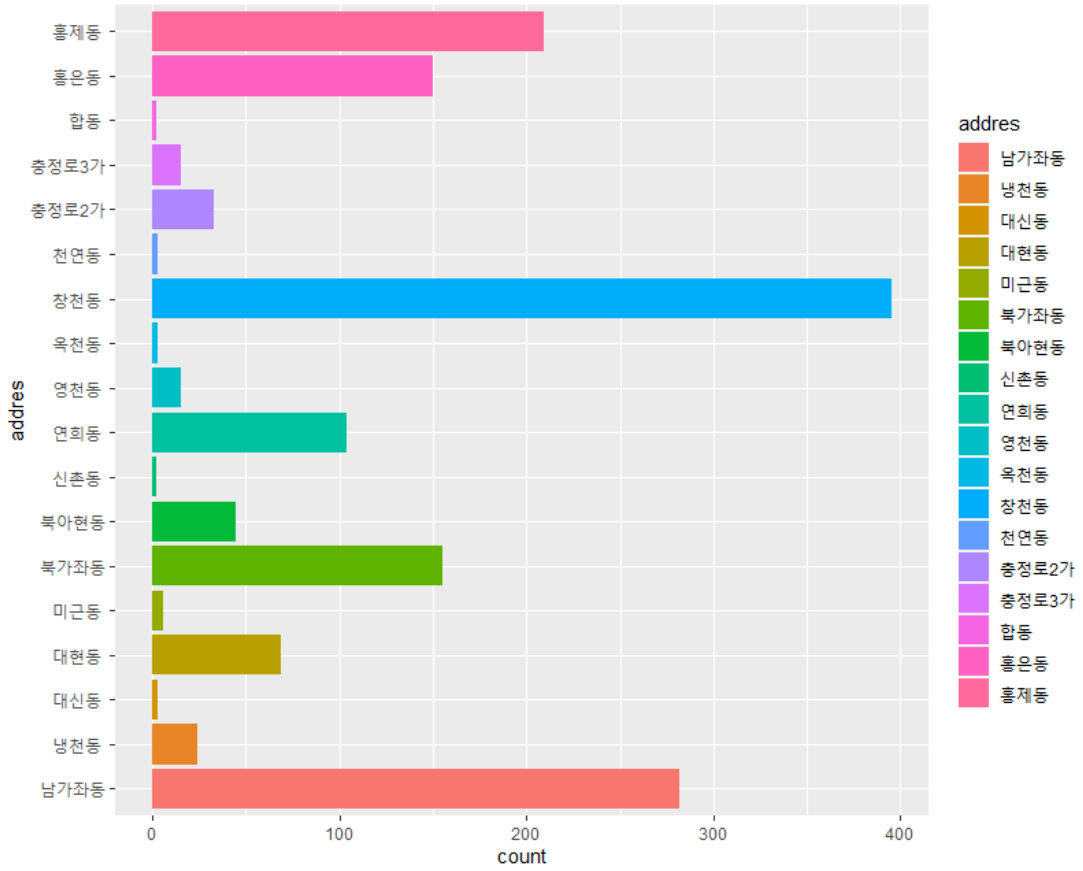

3e+06
2e+06
1e+06
0e+00
meanSalry
1~3년미만
10년이상
1년미만
3~5년미만
5~10년미만
Career classification
Career classification
1~3년미만
10년이상
1년미만
3~5년미만
5~10년미만

#--간단한 예제 2

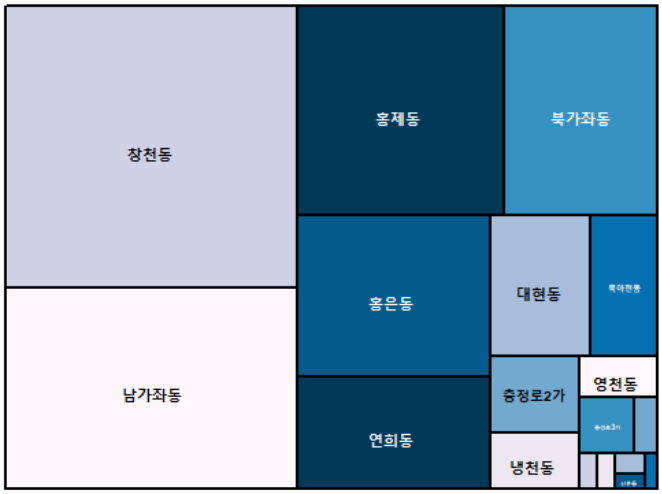
```r
# 2차실습
install.packages("readxl")
library(readxl)
ck <- read_xlsx(file.choose())
head(ck)



pattern <- regexpr("[가-힣]{1,4}동",ck$소재지전체주소)
dong <- regmatches(ck$소재지전체주소,pattern)
dong

# 실행결과 동 이름이 3글자인 경우와 4글자인 경우가 있으므로 지정한 자리만큼
# 글자를 추출하면 3글자인 동은 숫자가 포함된다.
# 공백과 숫자를 제거하자


a<- substr(ck$소재지전체주소,start =12, stop =16)

#첫번째 방법 - 근데이건 충청로 2가에서 2라는 숫자를 지워서 안된다. p259 쪽
dong <- gsub("[0-9]$¦[[:blank:]]", "" , a)

#두번째 방법
dong<-sub(pattern = "[0-9]+$¦ +$", # sub 로 $표현식(끝부분)으로 공백인 부분과 숫자인 부분을 뺀다
       replacement = "",
       x = a) # 정규표현식을 이용해서 모든숫자를 제거한다.


```

```r
# 동별 도수분포표 만들어보기
# table() 함수를 이용해서 숫자 세기, 변수가 한개일때 도수분표표를 만들어줌
# 도수분포표란 항목별 개수를 나타낸 것이다

#첫번째 방법
dong<- data.frame("dong" = dong) #일단 데이터 프레임을 만들었다. 그래야 타입이 생기고
str(dong)
class(dong)

dong$dong <- factor(dong$dong) #생긴 타입을 factor로 바꿔줬다.
dong
levels(dong$dong)          # 그러면 레벨에도 분류가 되고
addres<-data.frame(table(dong$dong)) # 여기서 테이블로 했을때 카운트를 넣을 수 있기 때문이다. 이상태로 데이터 프레임을 다시 만들었다.

names(addres) <- c('addres','count') # 이름을 새로 지정해주고
addres


#두번째 방법
addres2<-d %>% group_by(dong) %>%
  summarise(count=n())

addres2
str(addres2)

install.packages("dplyr")
library(dplyr)


ggplot(addres, aes(x=addres,y=count,fill=addres ))+   # 여기서 ggplot으로 그림을 그린다.
  geom_col() +
  coord_flip()

```

```r
# 트리맵은 옵션으로 데이터 프레임을 입력받는다.
# treemap(데이터, index=구분 열 , vSize=분포 열 , vColor=컬러, title=제목
library(treemapify)
library(ggplot2)


install.packages("treemap")
library(treemap)

# 여기서 index는 x축 처럼 넣을 값들 vsize는 개수를 나타낸다. 개수에 따라서 사이즈가 달라진다. vColr는 fill과 같고 마지막 title로 마무리한다.
treemap(addres, index="addres" , vSize="count" , vColor="addres", title="동별 통닭집 개수", palette="PuBu")
?palette

```


홍제동
홍은동
합동
충정로3가
충정로2가
천연동
창천동
옥천동
영천동
연희동
신촌동
북아현동
북가좌동
미근동
대현동
대신동
냉천동
남가좌동
addres
0
100
200
300
400
count
addres
남가좌동
냉천동
대신동
대현동
미근동
북가좌동
북아현동
신촌동
연희동
영천동
옥천동
창천동
천연동
충정로2가
충정로3가
합동
홍은동
홍제동

동별 통닭집 개수


창천동
홍제동
북가좌동
홍은동
대현동
남가좌동
충정로2가
영천동
연희동
냉천동

'R' 카테고리의 다른 글□

[R] 같은 형태의 ggplot 과 barplot 만들기 (차이 비교)□

[R] ggplot2 패키지 설치 에러시 해결 방법□

[R] R 을 활용한 데이터 탐색(Exploratory Data Analysis)□

[R] R ggplot 사용법 (데이터 시각화 도구)□

[R] R 에서 사용되는 기본적인 시각화 그래프-2□

[R] R 에서 사용되는 기본적인 시각화 그래프□

eda Exploratory Data Analysis ploratory Data Analysis R을 통한 데이터 탐색 R을 활용한 데이터 탐색 데이터 탐색

꾸까꾸

혼자 끄적끄적하는 블로그 입니다.