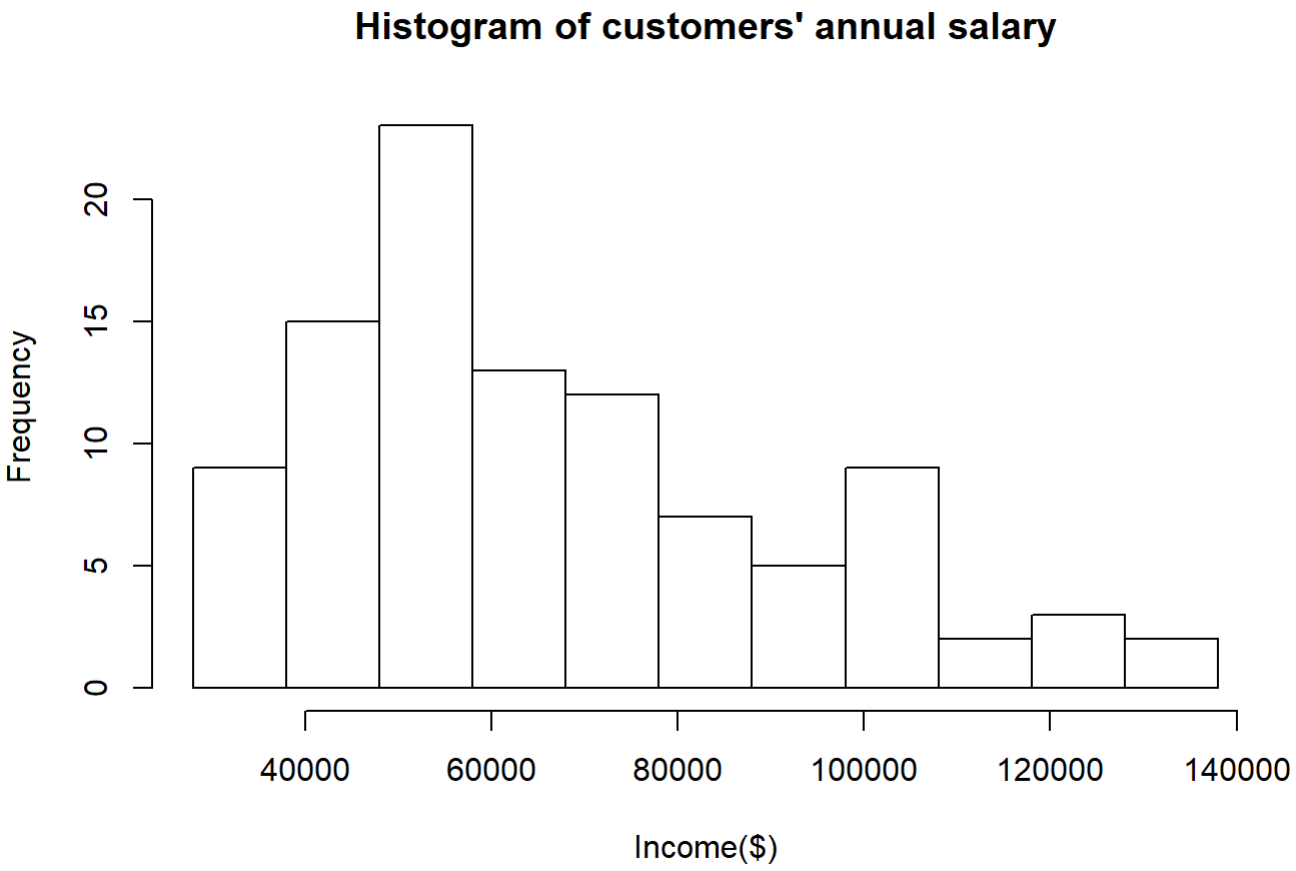
# Task 2 Basic predictive modelling

## 2.1 Identify the annual salary for each customer

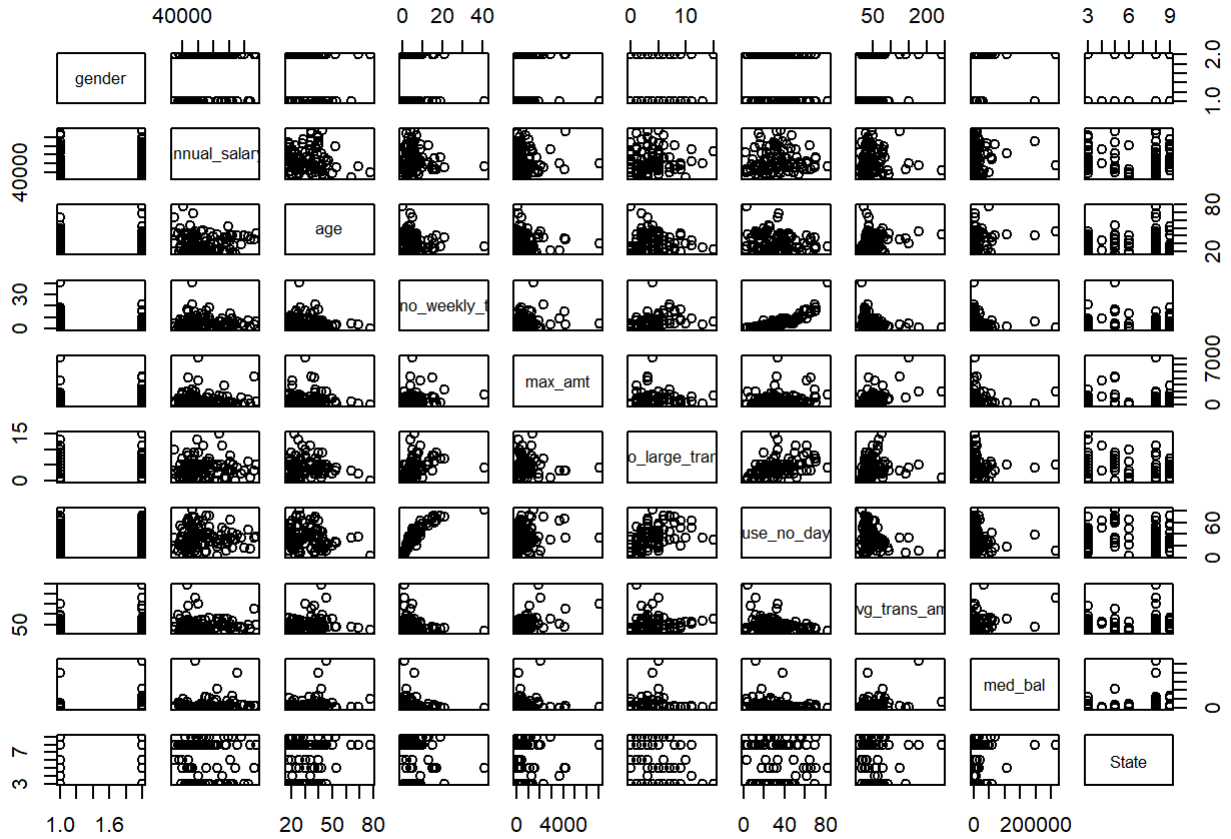
```
# firstly check the salary payment frequency of each customer
df_inc = data.frame(customer_id= unique(df_csmp$customer_id)) #create a data frame to store resu
lt

# create a mode function that will be used to find out what is the salary payment frequency
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# loop through all salary payment for each customer
# assume the salary level is constant for each customer over the observed period
for (i in seq(nrow(df_inc))){
  trans_data <- df[df$customer_id == df_inc$customer_id[i]
                   & df$txn_description=='PAY/SALARY',c("amount","date")] %>%
                group_by(date) %>%
               summarise(amount = sum(amount))

  total_s <- sum(trans_data$amount)
  count = dim(trans_data)[1]
  if ( count == 0){
    df_inc$freq[i] = NA
    df_inc$level[i] = NA
  } else {
    s=c()
    lvl = c()
    for (j in seq(count-1)){
      s = c(s,(trans_data$date[j+1]-trans_data$date[j]))
      lvl = c(lvl,trans_data$amount[j])
    }
    lvl = c(lvl,tail(trans_data$amount,n=1))
    df_inc$freq[i] = Mode(s)
    df_inc$level[i] = Mode(lvl)
  }
}

df_inc$annual_salary= df_inc$level / df_inc$freq *365.25

# visualise the distribution of customers' annual salary
hist(df_inc$annual_salary[!is.na(df_inc$annual_salary)],breaks=c(seq(28000,140000,by = 10000)),
     main = "Histogram of customers' annual salary", xlab= 'Income($)')
```




2.2 Explore correlations between annual salary and various customer attributes (e.g. age).

```
# create a dataframe to store relevant features for customers

df_cus <-df_csmp %>%  # use df_csmp to summarize customers' consumption behavior
  select (customer_id,gender,age,amount,date,balance) %>%
  group_by(customer_id) %>%
  mutate(avg_no_weekly_trans= round(7*n()/length(unique(df$date)),0),max_amt = max(amount),
         no_large_trans = sum(amount>100), # an arbitrary $100 benchmark is selected based on th
e
                                           # transaction amount histogram created in task 1.3
         use_no_day=length(unique(date)),
         avg_trans_amt = mean(amount, na.rm =TRUE),
         med_bal = median(balance,na.rm=TRUE)) %>%
  select(-c("amount","date","balance")) %>%
  unique()

# create additional features
df_cus$age_below20 <- ifelse(df_cus$age<20,1,0)
df_cus$age_btw20n40 <- ifelse(df_cus$age>=20 & df_cus$age <40,1,0)
df_cus$age_btw40n60 <- ifelse(df_cus$age>=40 & df_cus$age <60,1,0)

# investigate the state where customers live
# assume they live where most transactions occured (indicated by merchant_state)
df_region <-df_csmp %>%
  group_by(customer_id,merchant_state) %>%
  summarize(trans_count=n()) %>%
  group_by(customer_id) %>%
  mutate (no_state = n()) %>%
  filter(trans_count == max(trans_count))

# For equal number of transactions between multiple States, pick the most likely State
n_occur = data.frame(table(df_region$customer_id))
cus_id_rep = n_occur$Var1[n_occur$Freq > 1]

state_by_cust_no <- rev(names(sort(table(df_region$merchant_state),rev = TRUE)))
t = data.frame(customer_id = cus_id_rep, merchant_state=NA)

for (i in seq(length(cus_id_rep))){
  s = df_region$merchant_state[df_region$customer_id == cus_id_rep[i]]
  for (state in state_by_cust_no){
    if (state %in% s){
      t[i,2] = state
      break
    }
  }
}

df_region <- df_region[!(df_region$customer_id %in% cus_id_rep), c(1,2)] %>%
              as.data.frame() %>%
              rbind(t) %>%
              rename( State = merchant_state)

# merge all the features into single dataframe
df_cus <- df_cus %>% merge(df_inc) %>%
```

```
                  merge(df_region)

# extract relevant features
df_cus_attr <- df_cus %>%
               select("gender","annual_salary","age","avg_no_weekly_trans","max_amt",
                      "no_large_trans", "use_no_day","avg_trans_amt","med_bal","State")
plot(df_cus_attr)
```



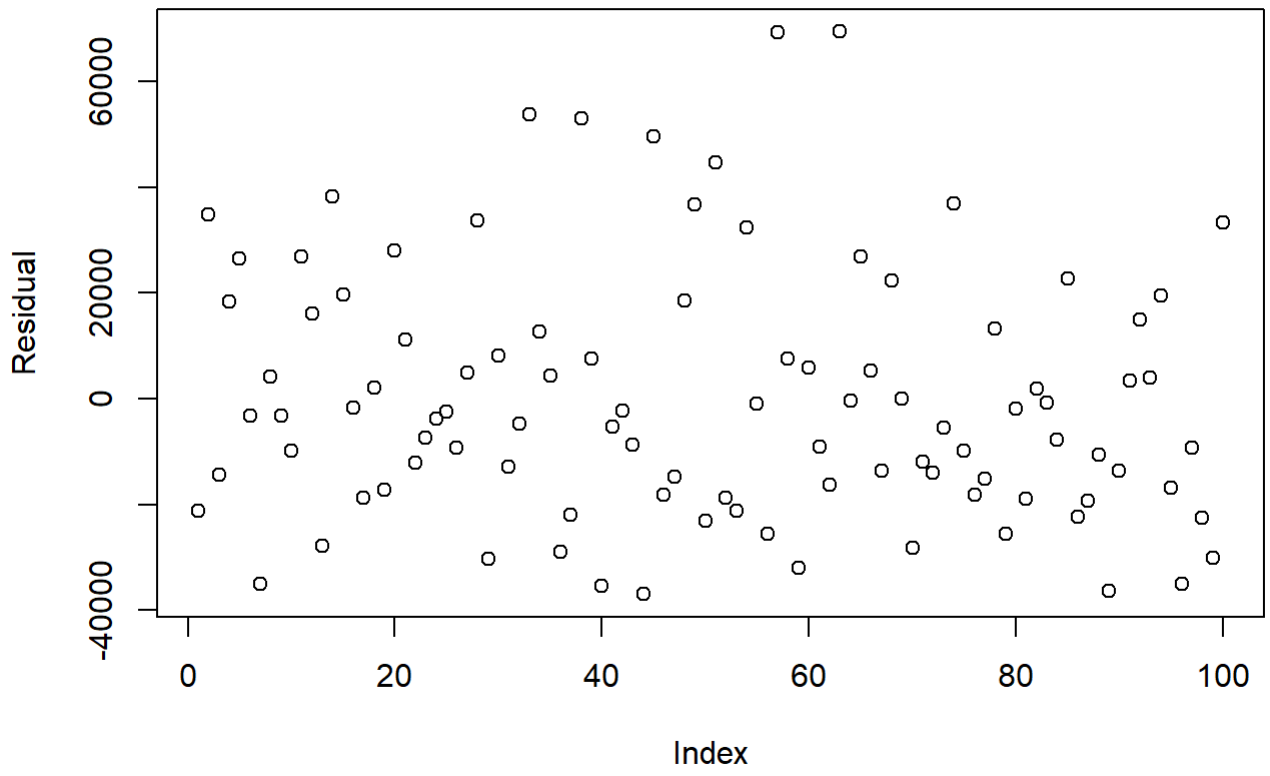

## 2.3 Build a simple regression model to predict the annual salary for each customer

```
# start with the model that includes all features created
fit1 <- lm(annual_salary ~.-customer_id - level-freq,data=df_cus)
summary(fit1)
MASS::stepAIC(fit1)
```

```
# backwards model selection, together with stepAIC yield the most appropriate model:
fit2 <- lm(formula = annual_salary ~ age + avg_trans_amt + med_bal +
    age_below20 + age_btw20n40 + age_btw40n60, data = df_cus)


summary(fit2)
```

```
##
## Call:
## lm(formula = annual_salary ~ age + avg_trans_amt + med_bal +
##     age_below20 + age_btw20n40 + age_btw40n60, data = df_cus)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -36951 -17459  -3556  13641  69382
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -8.008e+03  3.555e+04  -0.225   0.8223
## age            7.136e+02  4.688e+02   1.522   0.1313
## avg_trans_amt -1.142e+02  8.189e+01  -1.395   0.1664
## med_bal        1.625e-01  7.758e-02   2.095   0.0389 *
## age_below20    6.711e+04  2.931e+04   2.290   0.0243 *
## age_btw20n40   5.900e+04  2.486e+04   2.373   0.0197 *
## age_btw40n60   4.621e+04  2.013e+04   2.296   0.0239 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24650 on 93 degrees of freedom
## Multiple R-squared:  0.0992, Adjusted R-squared:  0.04109
## F-statistic: 1.707 on 6 and 93 DF,  p-value: 0.128
```

```
# examine the residules to capture any missed relationships
plot(fit2$residuals, ylab = 'Residual')
```




## 2.4 How accurate is your model?

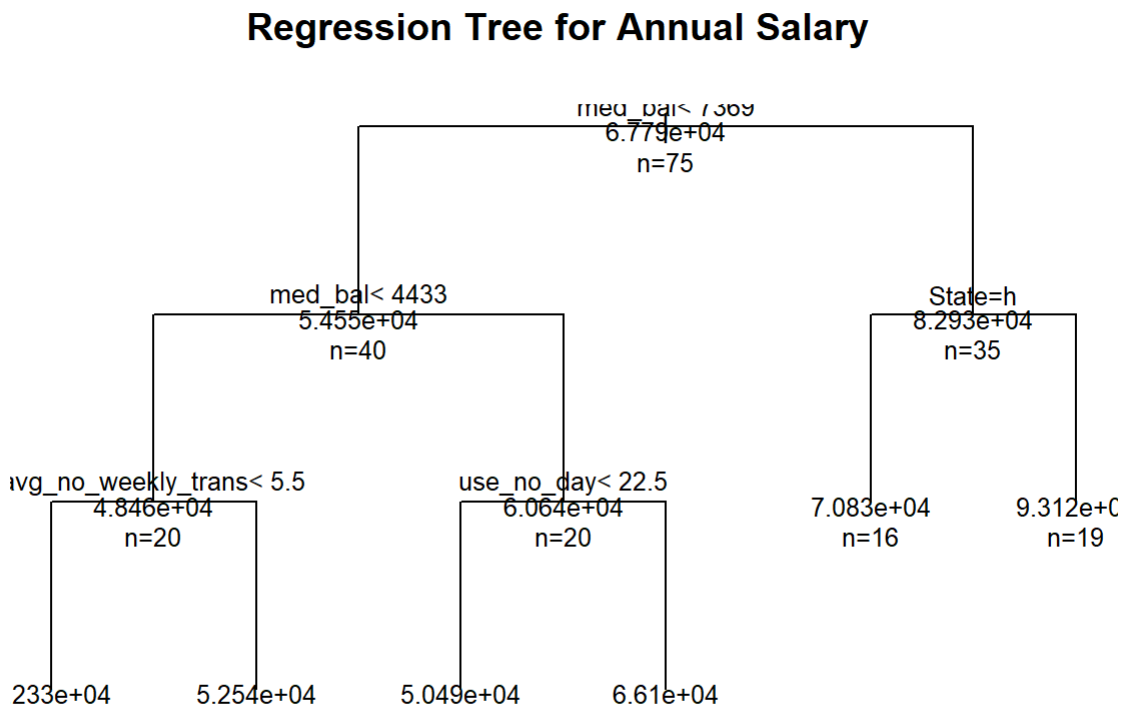
```
# to see model accuracy
rmse(fit2,df_cus)
```

```
## [1] 23767.49
```

The RMSE of the model over the whole dataset is over 20000, which indicates the inaccuracy of the model.The model's adjusted R^2 also shows that it only explains about 4% of variation in customers'annual salary. It is thus risky to use this linear model to predict customers' income bracket. More data is required to develop a more reliable model.

## 2.5 For a challenge: build a decision-tree based model to predict salary.

```
# split into train and test datasets
smp_size <- floor(0.75 * nrow(df_cus))

## set the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(df_cus)), size = smp_size)

df_cus_train <- df_cus[train_ind, ]
df_cus_test <- df_cus[-train_ind, ]

fit3 <- rpart(annual_salary ~gender + age + avg_no_weekly_trans +  max_amt + no_large_trans + us
e_no_day + avg_trans_amt + med_bal + age_below20 + age_btw20n40 + age_btw40n60 + State, method=
"anova", data=df_cus_train)

# plot tree
plot(fit3, uniform=TRUE,
   main="Regression Tree for Annual Salary ")
text(fit3, use.n=TRUE, all=TRUE, cex=.8)
```




```
# examine the prediction accuracy
rmse(fit3, df_cus_test)
```

```
## [1] 25672.77
```