**2019.06.26**

**Final Project**

**기술문서**

**김영욱 신동준 정진욱 박종민 임종윤 최수인**

1. **개요**

KCB, DACON에서 제공하는 제주도민의 금융정보 데이터셋을 표본집단으로 활용해 사용자가 입력하는 금융정보를 통해 신용등급을 예측하는 어플리케이션 개발을 목적으로 함.

데이터셋의 크기가 HDFS 환경에 저장할 정도로 충분히 크지는 않지만 제공받은 데이터와 Flask로 구현한 웹상에서의 사용자 입력값을 Flume, Flask, Sqoop을 통해 HDFS에 축적,

SparkR을 이용해 통계적인 지식을 동원하여 데이터셋의 분석, 예측 및 시각화를 하고자 한다.

1. **실행환경**

OS : Oracle VMware - Centos6

WEB : Python3.6 – Flask

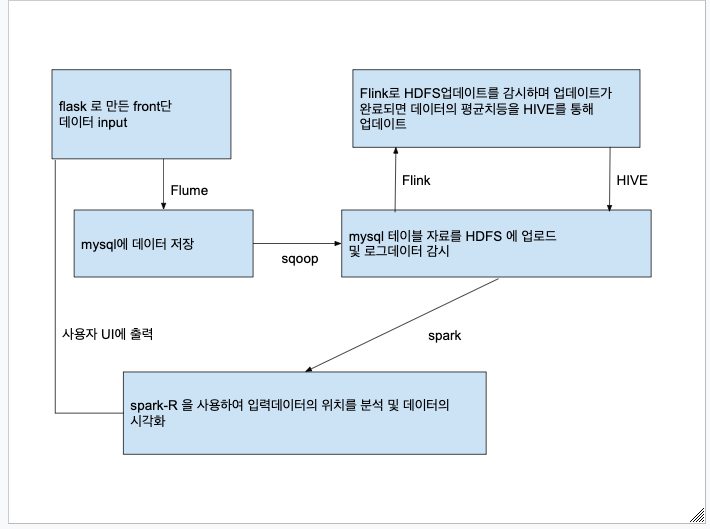
DBMS : MySql-5.1.73

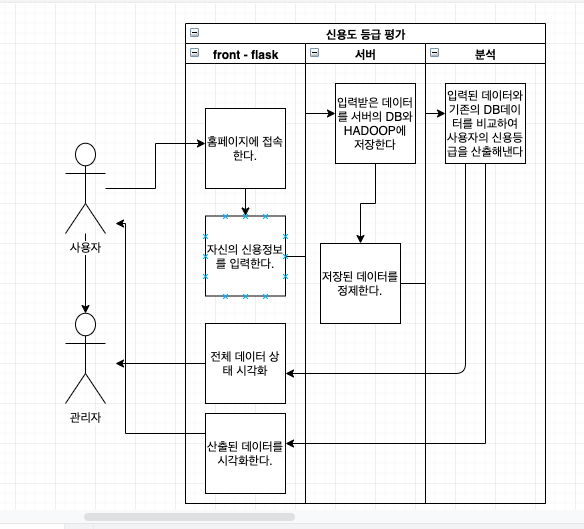
Data Streaming : Flume, Sqoop

DFS : Hadoop File System

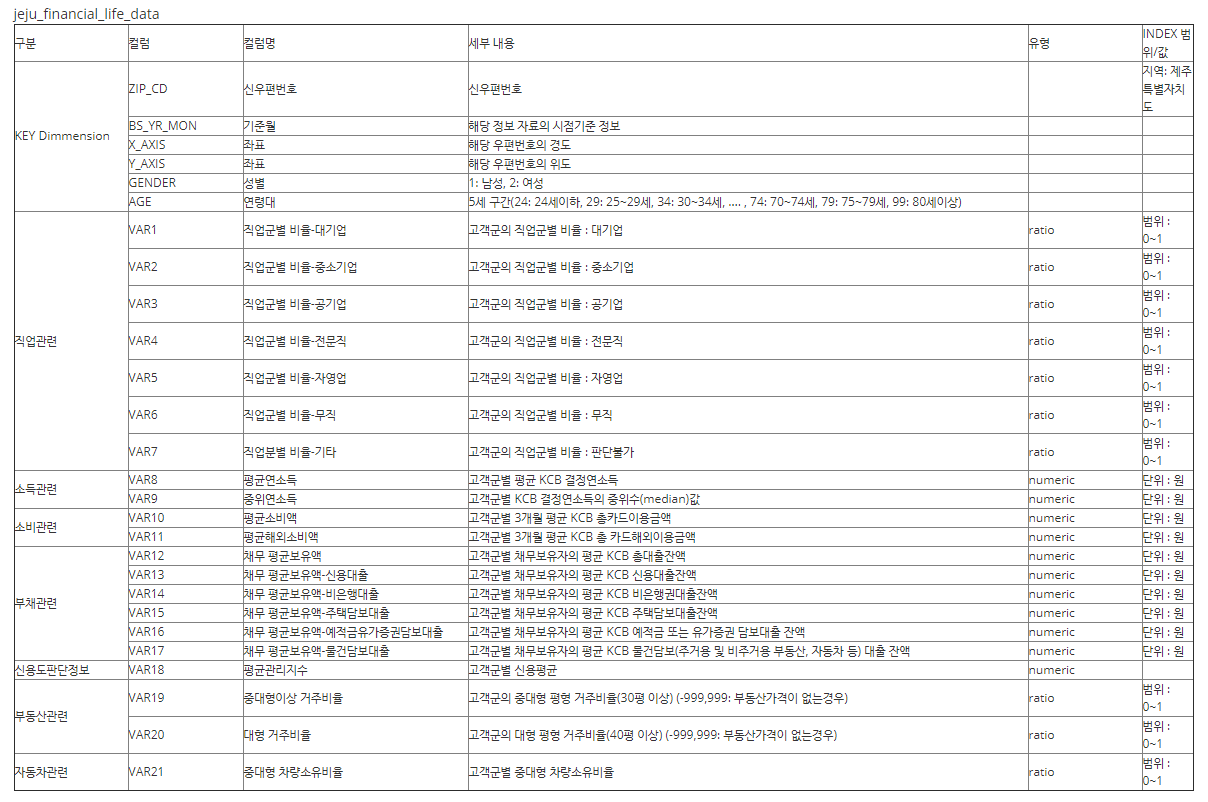
Statistic Tool : SparkR,

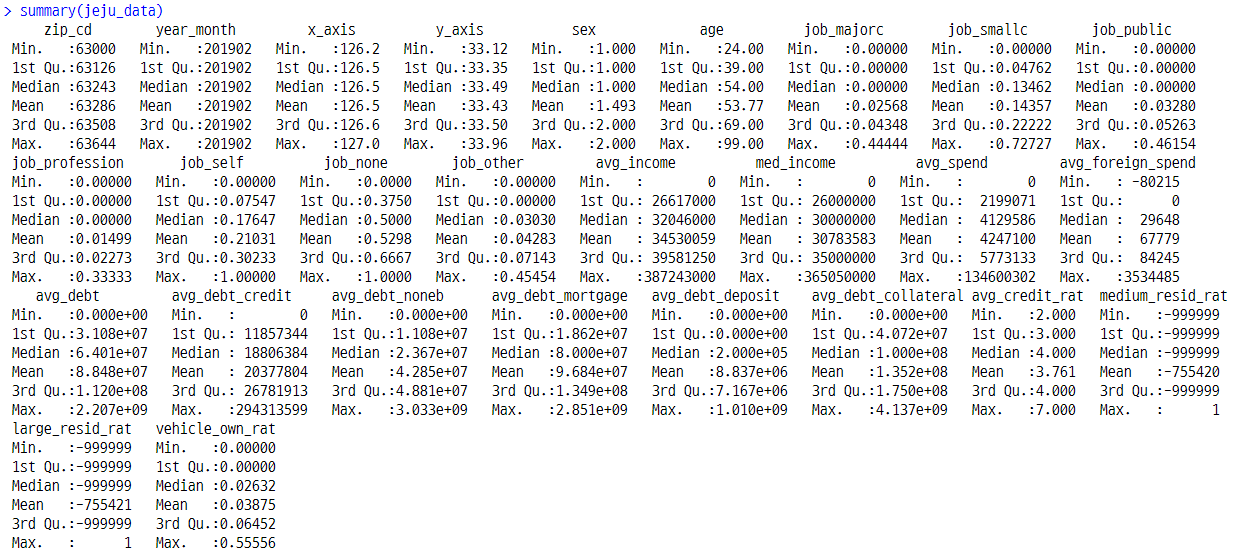
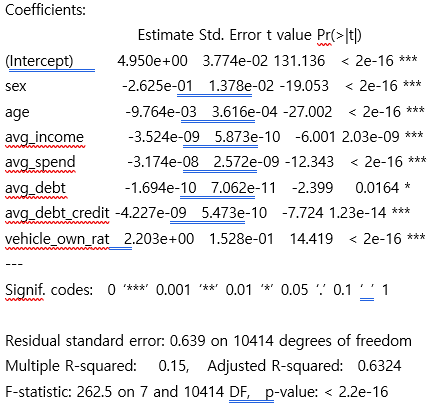
1. **시스템 순서도 및 유즈케이스 다이어그램**





1. 데이터 분석 및 결측갑 제거



1. **응용기술**

**3-1 Hadoop설정**

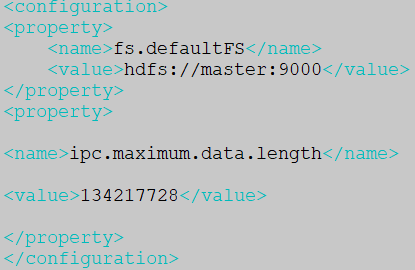


그림 <core-site.xml>



그림 <hdfs-site.xml>

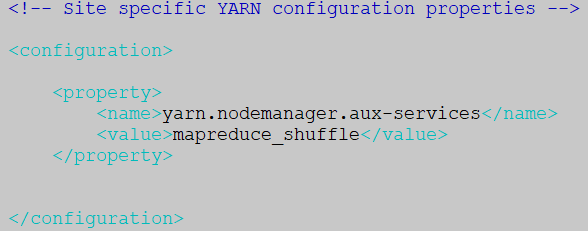


그림 <yarn-site.xml>

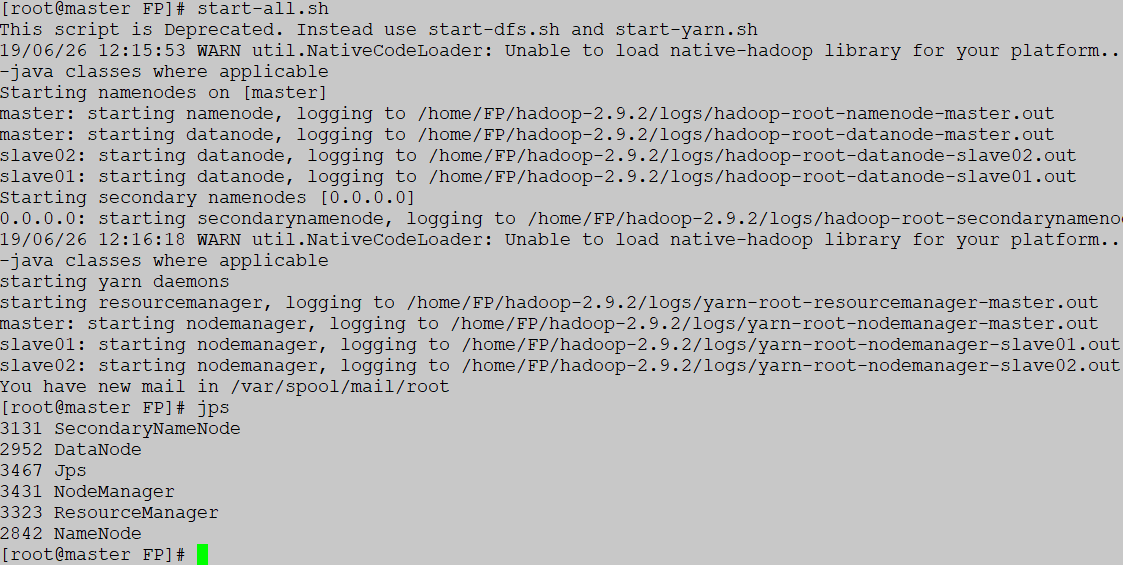
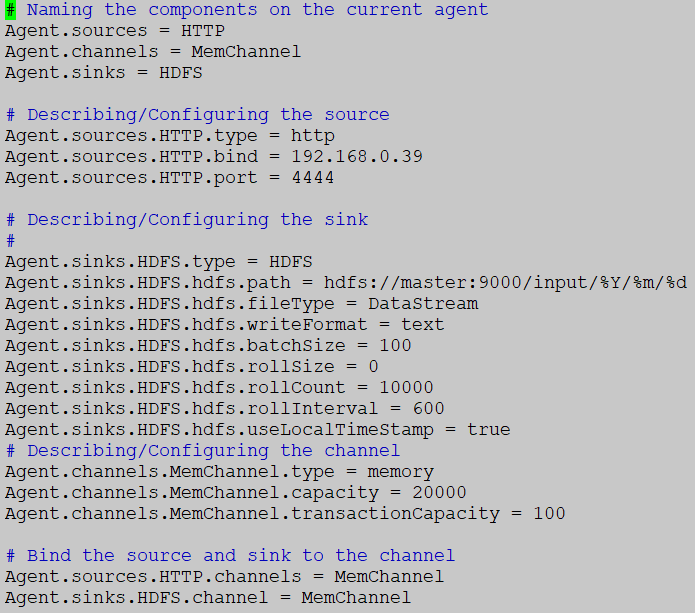
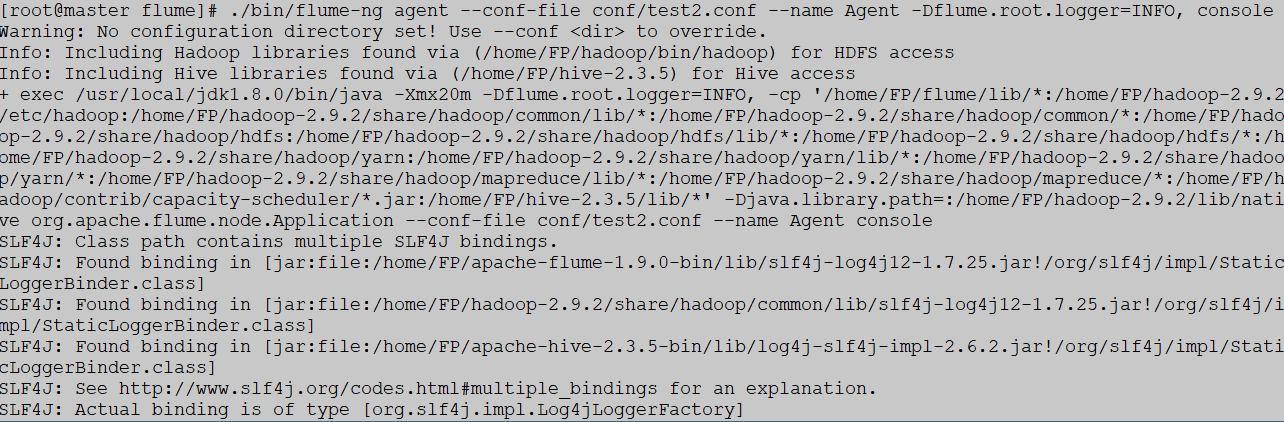


그림 <start-all.sh>

**3-2 flume설정**





**4444포트에 request를 전송하면 Flume이 감지하여 hadoop에 데이터를 날짜형식에 맞춰 저장한다.**

**3-3 mysql 금융 데이터셋 테이블**

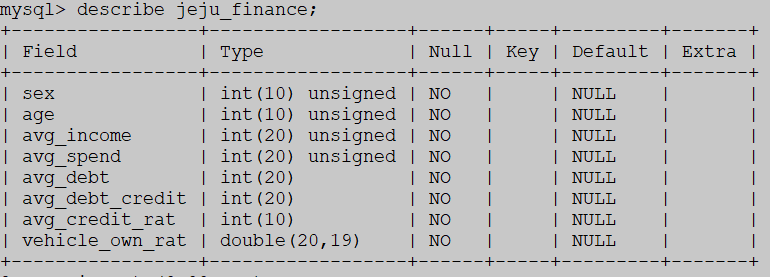


그림 <describe jeju\_finance>

실제 데이터셋의 칼럼은 28개이지만 skeness,, kurtosis를 R을 통해 분석하여 유의미한 칼럼만 추출하여 사용하였다.

**3-4 Python Flask**

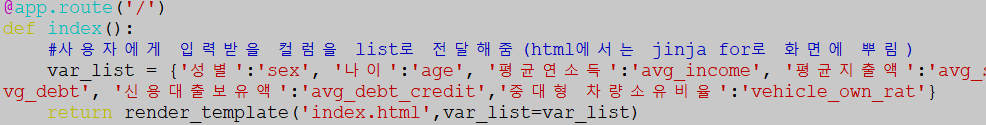


그림 <index>

사용자에게 입력 받을 정보를 Dictionary로 전달

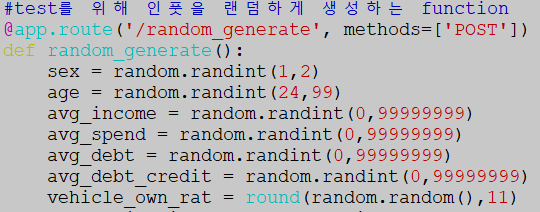


그림 <random\_generate>

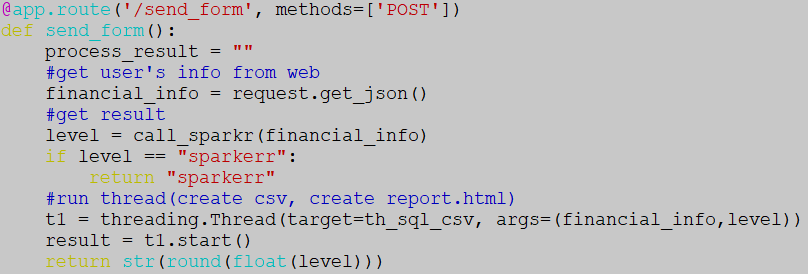


그림 <입력값 전송시 동작하는 함수>

사용자가 입력한 값을 financial\_info에 저장후,

Call\_넴갂로 SparkRR을 호출해 신용등급 분석 결과를 리턴

결과가 리턴되면 hdfs file system, lcoal상의 csv로 기록하는 스레드를 호출한다

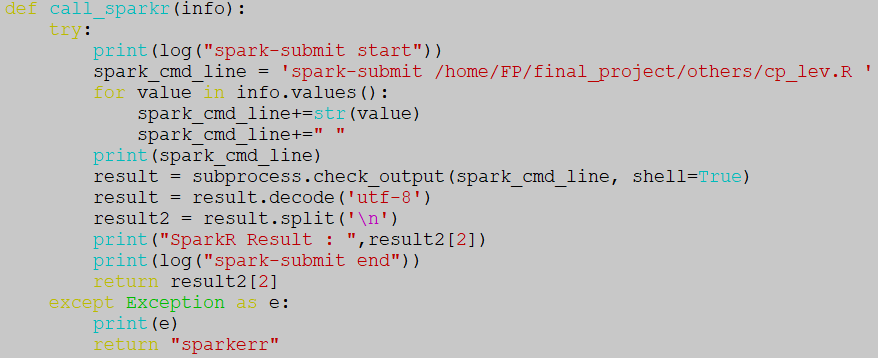


그림 <call\_sparkr>

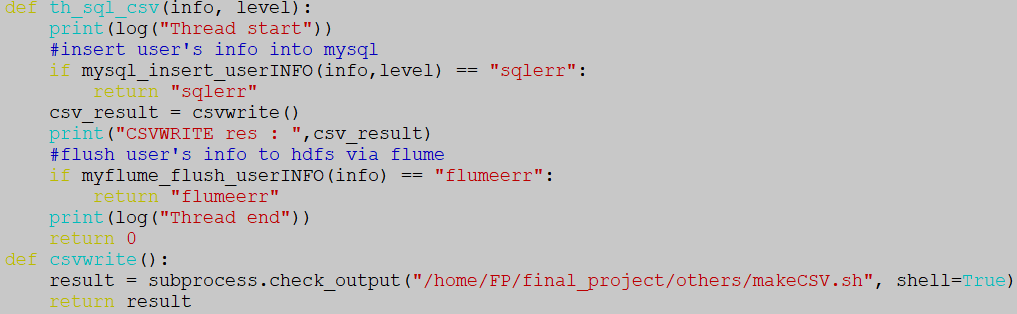


그림 <스레드가 호출하는 함수>

.

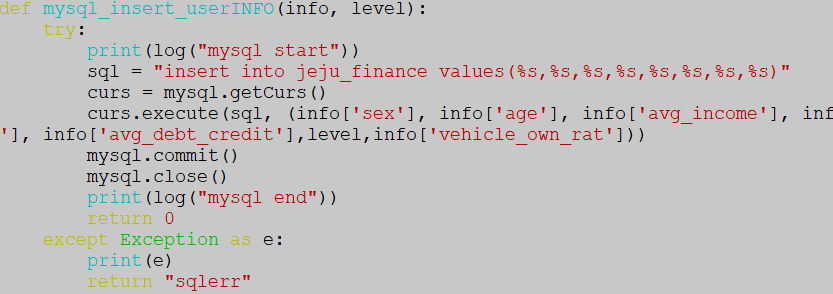


그림 <sql-flask연동>

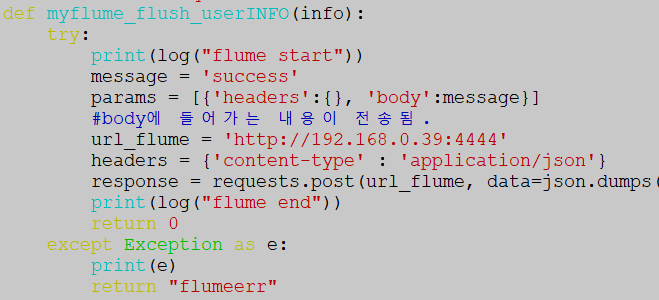
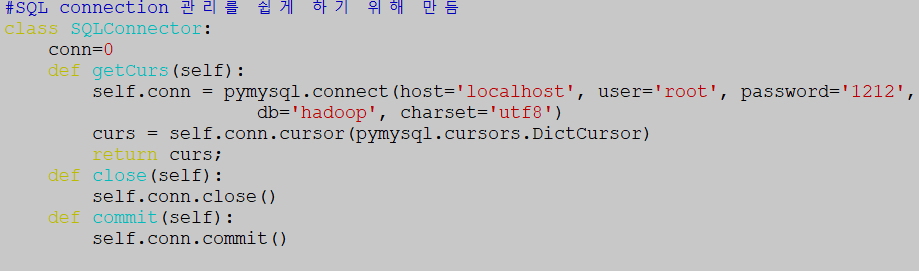


그림 <flask-flume연동>

**3-5 mysql connection**



**3-6 Rscript, ShellScript구현**

사용자가 입력값을 전송 후 일어나는 핵심 동작들을 스크립트로 만들어서 Flask내에서 subprocess로 호출한다.

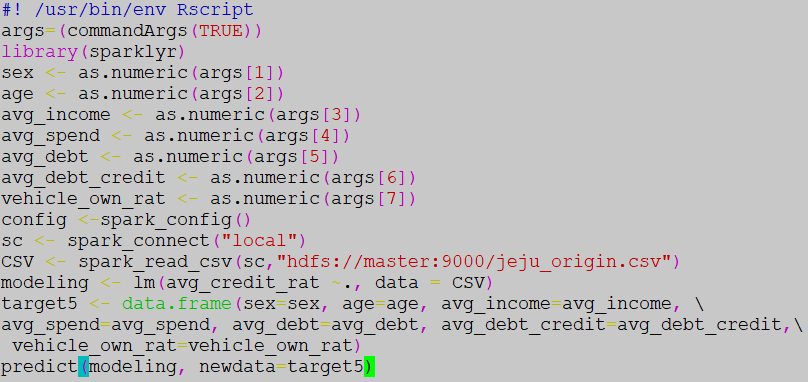


그림 <cp\_lev.R>

사용자의 신용등급을 예측 및 분석하기 위한 RScript이다.

매개변수 형식으로 입력값을 전달 받고

Sparklyr 패키지 내에 있는 Spark\_read\_csv 함수를 사용해 HDFS의 기존 데이터를 데이터프레임으로 가져온다.

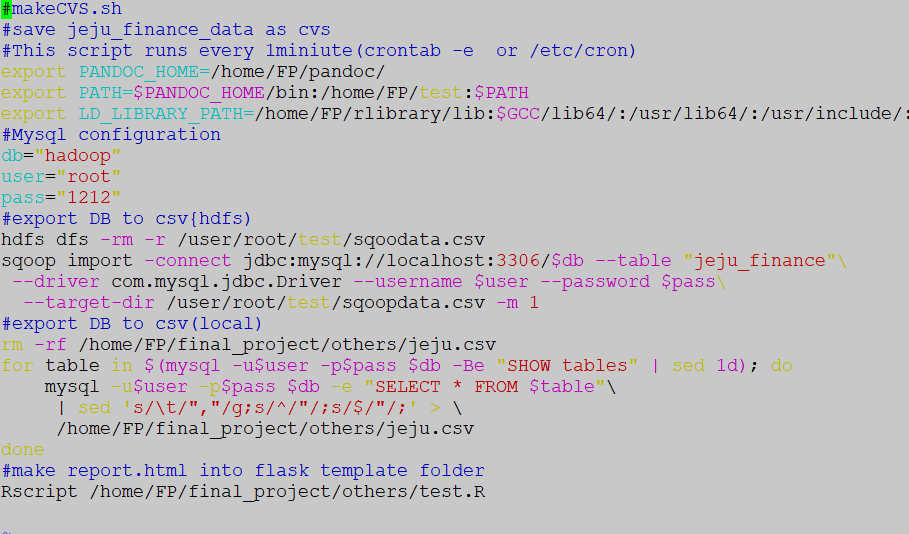
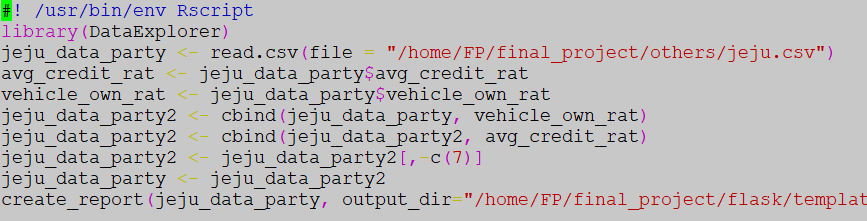


그림 <makeCVS.sh>

주석에 나와있듯이 sql 로그인 정보를 정의하고 hdfs에 저장, sqoop import를 사용해 DB의 내용을 하둡에 저장, 마지막으로 로컬 파일 시스템에 저장한다.

그리고 나서 웹에서 제공하고 있는 현재 데이터의 시각화정보를 갱신하기 위해 test.R을 호출한다.

특히, test.R에서 수행되는 create\_report가 다소 오래 걸리기 때문에 Flask에서 스레드로 구현하였다. 그래서 웹에서 새로운 row가 반영되는 데에는 약 20초 정도가 소요된다.



**3-7 html 구현**

**<index.html>**

|  |  |
| --- | --- |
|  | <!DOCTYPE html> |
|  | <html lang="en"> |
|  | <head> |
|  | <title>index</title> |
|  | <meta charset="utf-8"> |
|  | <meta name="viewport" content="width=device-width, initial-scale=1"> |
|  | <link rel="stylesheet" href="<https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css>"> |
|  | <script src="<https://ajax.googleapis.com/ajax/libs/jquery/3.4.0/jquery.min.js>"></script> |
|  | <script src="<https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js>"></script> |
|  | </head> |
|  | <body> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <div class="container"> |
|  | <h2>index Page</h2> |
|  | <p>hello~</p> |
|  |  |
|  | <ul class="nav nav-tabs"> |
|  | <li class="active"><a data-toggle="tab" href="[#home](file:///C:\Users\kim00\Desktop\templates\index.html#home)">Home</a></li> |
|  | <li><a data-toggle="tab" href="[#menu1](file:///C:\Users\kim00\Desktop\templates\index.html#menu1)">신용등급 확인</a></li> |
|  | <li><a data-toggle="tab" href="[#menu2](file:///C:\Users\kim00\Desktop\templates\index.html#menu2)">report</a></li> |
|  | <li><a data-toggle="tab" href="[#menu3](file:///C:\Users\kim00\Desktop\templates\index.html#menu3)">회사소개</a></li> |
|  | </ul> |
|  |  |
|  | <div class="tab-content"> |
|  | <div id="home" class="tab-pane fade in active"> |
|  | <h3>HOME</h3> |
|  | <p>hello~</p> |
|  | </div> |
|  |  |
|  | <div id="menu1" class="tab-pane fade"> |
|  | <h3>신용등급 확인</h3> |
|  | <p>hello</p> |
|  | <div class="col-md-6 col-md-offset-3"> |
|  | <h3>금융정보 입력폼</h3> |
|  | </div> |
|  | <div class="col-sm-6 col-md-offset-3"> |
|  | <li>{% for key,value in var\_list.items() %}</li> |
|  | <li class="list-group-item list-group-item-info"> {{key}} </li> |
|  | <li class="list-group-item list-group-item-danger"> |
|  | {% if loop.index ==1 %} |
|  | <label><input type="radio" id="{{value}}" name="sex\_radio" class="var\_sex", value=1>남자</label> |
|  | <label><input type="radio" id="{{value}}" name="sex\_radio" class="var\_sex", value=2>여자</label></li> |
|  | {% else %} |
|  | <input type="text" id="{{value}}" name="{{value}}" class="var\_list"></li> |
|  | {% endif %} |
|  | <li class="list-group-item list-group-item-danger"> |
|  |  |
|  | {% endfor %}</li> |
|  | <div class="form-group text-center"> |
|  | <button type="button" class ="btn btn-info" onclick="random\_generate()">랜덤생성</button> |
|  | <button type="button" class="btn btn-info" onclick="send\_form()">제출</button> |
|  | </div></div> |
|  | <div><img src="" id="grade\_show" style="width: 100px; height:120px;"></div> |
|  | </div> |
|  |  |
|  | <div id="menu2" class="tab-pane fade"> |
|  | <h3>?</h3> |
|  | <p>hello</p> |
|  | <button id="movereport" class="btn btn-info">report.html</button> |
|  | </div> |
|  |  |
|  | <div id="menu3" class="tab-pane fade"> |
|  | <h3>회사소개</h3> |
|  | <p>회사소개</p> |
|  | </div> |
|  | </div> |
|  |  |
|  | </body> |
|  | <script src="<https://code.jquery.com/jquery-3.4.1.min.js>"></script> |
|  | <script> |
|  | $('#movereport').click(function(){ |
|  | location.href="/report\_view"; |
|  | }) |
|  | var success = 0; |
|  | function random\_generate(){ |
|  | $.ajax({ |
|  | url : "random\_generate", |
|  | method : "post", |
|  | type : "json", |
|  | contentType : "application/json", |
|  | success : function(data, status){ |
|  | var radio\_obj = document.getElementsByName('sex\_radio'); |
|  | for(i=0; i<radio\_obj.length; i++){ |
|  | if(radio\_obj[i].value == data[0]){ |
|  | radio\_obj[i].checked = true; |
|  | } |
|  | } |
|  | var list = $('.var\_list'); |
|  | for(var i = 0; i<list.length; i++){ |
|  | $(list[i]).val(data[i+1]) |
|  | } |
|  | } |
|  | }); |
|  | } |
|  | function send\_form(){ |
|  | var financial\_info = { |
|  | sex : 0, |
|  | age :0, |
|  | avg\_income:0, |
|  | avg\_spend:0, |
|  | avg\_debt:0, |
|  | avg\_debt\_credit:0, |
|  | vehicle\_own\_rat:0 |
|  |  |
|  | } |
|  | var obj = document.getElementsByName('sex\_radio'); |
|  | var checked\_index = -1; |
|  | var checked\_value = ''; |
|  | for(i=0; i<obj.length; i++){ |
|  | if(obj[i].checked){ |
|  | checked\_value=obj[i].value; |
|  | } |
|  | } |
|  | financial\_info.sex = checked\_value; |
|  | financial\_info.age = $('#age').val(); |
|  | financial\_info.avg\_income= $('#avg\_income').val(); |
|  | financial\_info.avg\_spend= $('#avg\_spend').val(); |
|  | financial\_info.avg\_debt= $('#avg\_debt').val(); |
|  | financial\_info.avg\_debt\_credit= $('#avg\_debt\_credit').val(); |
|  | financial\_info.vehicle\_own\_rat= $('#vehicle\_own\_rat').val(); |
|  | $.ajax({ |
|  | url:"send\_form", |
|  | method : "post", |
|  | type : "json", |
|  | contentType : "application/json", |
|  | data : JSON.stringify(financial\_info), |
|  | success : function(data){ |
|  | if(data == "sparkerr"){ |
|  | confirm("sparkerr") |
|  | } |
|  | else{ |
|  | confirm("전송되었습니다.") |
|  | console.log(data) |
|  | if(data < 1){ |
|  | $('#grade\_show').attr('src','/static/CSS/image/1.png'); |
|  | }else{ |
|  | $('#grade\_show').attr('src','/static/CSS/image/'+data+'.png'); |
|  | } |
|  | } |
|  | } |
|  | }); |
|  | } |
|  | </script> |
|  |  |
|  |  |
|  | </html> |
|  |  |

**<financial\_form.html>**

<!DOCTYPE html>

<html lang="ko">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>financial\_form.html</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.2/css/bootstrap.min.css">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.2/css/bootstrap-theme.min.css">

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.2/js/bootstrap.min.js"></script>

<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>

</head>

<body>

<article class="container">

</article>

</body>

<script>

</script>

# </html>

# ***4. R을 이용한 데이터셋 분석 코드***

파이널 프로젝트에서 사용되었던 코드들입니다.

##라이브러리 호출  
  
library(shiny)  
library(ggplot2)  
library(tidyverse)

## -- Attaching packages ---------------------------------------------------------------------------------------------------------------------- tidyverse 1.2.1 --

## √ tibble 2.1.3 √ purrr 0.3.2  
## √ tidyr 0.8.3 √ dplyr 0.8.1  
## √ readr 1.3.1 √ stringr 1.4.0  
## √ tibble 2.1.3 √ forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

#library(tree)  
library(rpart)  
library(dplyr)  
library(neuralnet)

##   
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':  
##   
## compute

library(nnet)  
library(caret)  
library(ROCR)

## Loading required package: gplots

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

##   
## Attaching package: 'ROCR'

## The following object is masked from 'package:neuralnet':  
##   
## prediction

library(devtools)  
library(NeuralNetTools)  
library(UsingR)

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

## Loading required package: HistData

## Loading required package: Hmisc

## Loading required package: survival

##   
## Attaching package: 'survival'

## The following object is masked from 'package:rpart':  
##   
## solder

## The following object is masked from 'package:caret':  
##   
## cluster

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

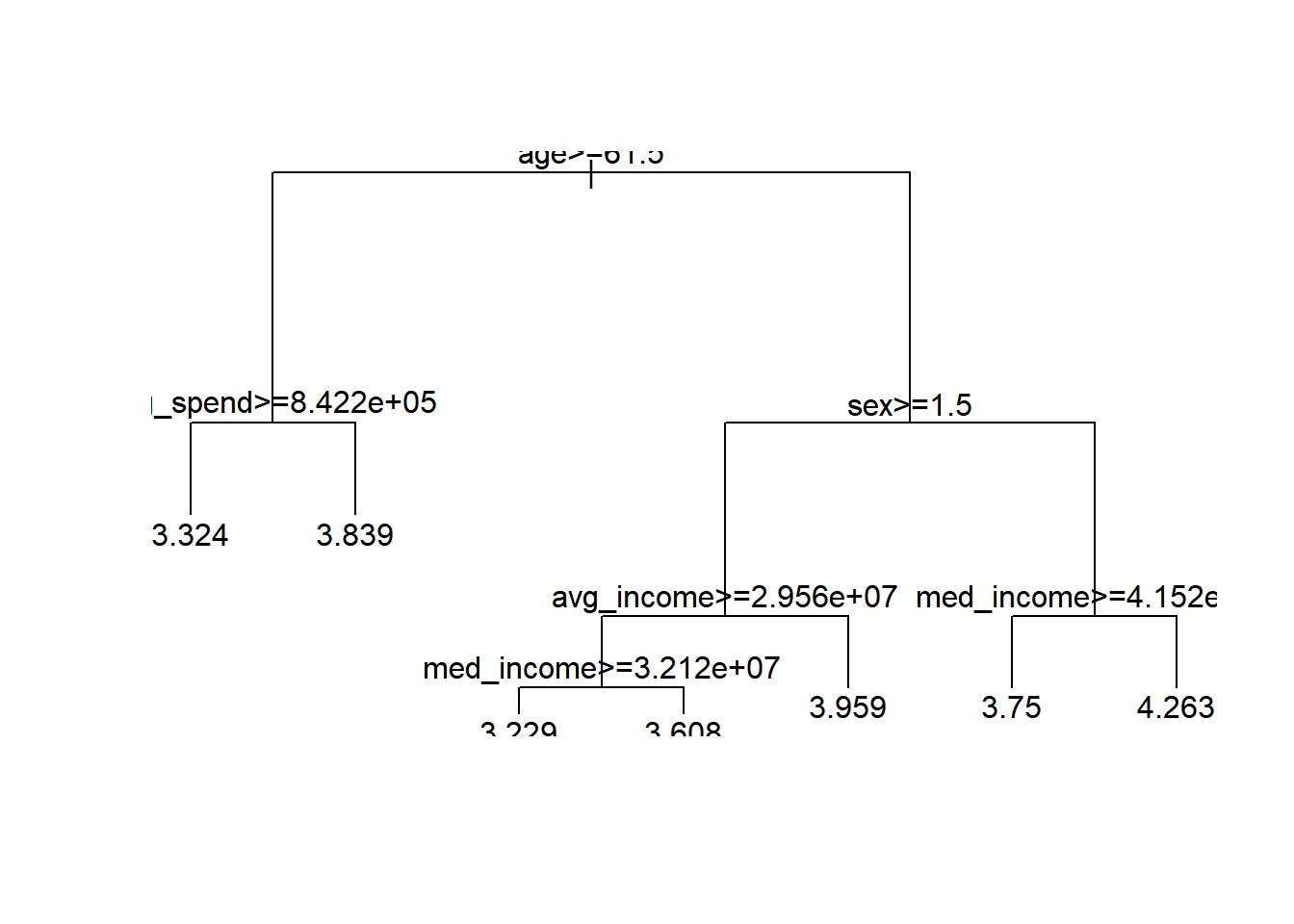
##   
## Attaching package: 'UsingR'

## The following object is masked from 'package:survival':  
##   
## cancer

jeju\_data <- read.csv(file = "C:/Users/daily/Desktop/FinalProject/FinalProject/jeju\_financial\_life\_data.csv")  
  
summary(jeju\_data)

## zip\_cd year\_month x\_axis y\_axis   
## Min. :63000 Min. :201902 Min. :126.2 Min. :33.12   
## 1st Qu.:63126 1st Qu.:201902 1st Qu.:126.5 1st Qu.:33.35   
## Median :63243 Median :201902 Median :126.5 Median :33.49   
## Mean :63286 Mean :201902 Mean :126.5 Mean :33.43   
## 3rd Qu.:63508 3rd Qu.:201902 3rd Qu.:126.6 3rd Qu.:33.50   
## Max. :63644 Max. :201902 Max. :127.0 Max. :33.96   
## sex age job\_majorc job\_smallc   
## Min. :1.000 Min. :24.00 Min. :0.00000 Min. :0.00000   
## 1st Qu.:1.000 1st Qu.:39.00 1st Qu.:0.00000 1st Qu.:0.04762   
## Median :1.000 Median :54.00 Median :0.00000 Median :0.13462   
## Mean :1.493 Mean :53.77 Mean :0.02568 Mean :0.14357   
## 3rd Qu.:2.000 3rd Qu.:69.00 3rd Qu.:0.04348 3rd Qu.:0.22222   
## Max. :2.000 Max. :99.00 Max. :0.44444 Max. :0.72727   
## job\_public job\_profession job\_self job\_none   
## Min. :0.00000 Min. :0.00000 Min. :0.00000 Min. :0.0000   
## 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.07547 1st Qu.:0.3750   
## Median :0.00000 Median :0.00000 Median :0.17647 Median :0.5000   
## Mean :0.03280 Mean :0.01499 Mean :0.21031 Mean :0.5298   
## 3rd Qu.:0.05263 3rd Qu.:0.02273 3rd Qu.:0.30233 3rd Qu.:0.6667   
## Max. :0.46154 Max. :0.33333 Max. :1.00000 Max. :1.0000   
## job\_other avg\_income med\_income   
## Min. :0.00000 Min. : 0 Min. : 0   
## 1st Qu.:0.00000 1st Qu.: 26617000 1st Qu.: 26000000   
## Median :0.03030 Median : 32046000 Median : 30000000   
## Mean :0.04283 Mean : 34530059 Mean : 30783583   
## 3rd Qu.:0.07143 3rd Qu.: 39581250 3rd Qu.: 35000000   
## Max. :0.45454 Max. :387243000 Max. :365050000   
## avg\_spend avg\_foreign\_spend avg\_debt   
## Min. : 0 Min. : -80215 Min. :0.000e+00   
## 1st Qu.: 2199071 1st Qu.: 0 1st Qu.:3.108e+07   
## Median : 4129586 Median : 29648 Median :6.401e+07   
## Mean : 4247100 Mean : 67779 Mean :8.848e+07   
## 3rd Qu.: 5773133 3rd Qu.: 84245 3rd Qu.:1.120e+08   
## Max. :134600302 Max. :3534485 Max. :2.207e+09   
## avg\_debt\_credit avg\_debt\_noneb avg\_debt\_mortgage   
## Min. : 0 Min. :0.000e+00 Min. :0.000e+00   
## 1st Qu.: 11857344 1st Qu.:1.108e+07 1st Qu.:1.862e+07   
## Median : 18806384 Median :2.367e+07 Median :8.000e+07   
## Mean : 20377804 Mean :4.285e+07 Mean :9.684e+07   
## 3rd Qu.: 26781913 3rd Qu.:4.881e+07 3rd Qu.:1.349e+08   
## Max. :294313599 Max. :3.033e+09 Max. :2.851e+09   
## avg\_debt\_deposit avg\_debt\_collateral avg\_credit\_rat medium\_resid\_rat   
## Min. :0.000e+00 Min. :0.000e+00 Min. :2.000 Min. :-999999   
## 1st Qu.:0.000e+00 1st Qu.:4.072e+07 1st Qu.:3.000 1st Qu.:-999999   
## Median :2.000e+05 Median :1.000e+08 Median :4.000 Median :-999999   
## Mean :8.837e+06 Mean :1.352e+08 Mean :3.761 Mean :-755420   
## 3rd Qu.:7.167e+06 3rd Qu.:1.750e+08 3rd Qu.:4.000 3rd Qu.:-999999   
## Max. :1.010e+09 Max. :4.137e+09 Max. :7.000 Max. : 1   
## large\_resid\_rat vehicle\_own\_rat   
## Min. :-999999 Min. :0.00000   
## 1st Qu.:-999999 1st Qu.:0.00000   
## Median :-999999 Median :0.02632   
## Mean :-755421 Mean :0.03875   
## 3rd Qu.:-999999 3rd Qu.:0.06452   
## Max. : 1 Max. :0.55556

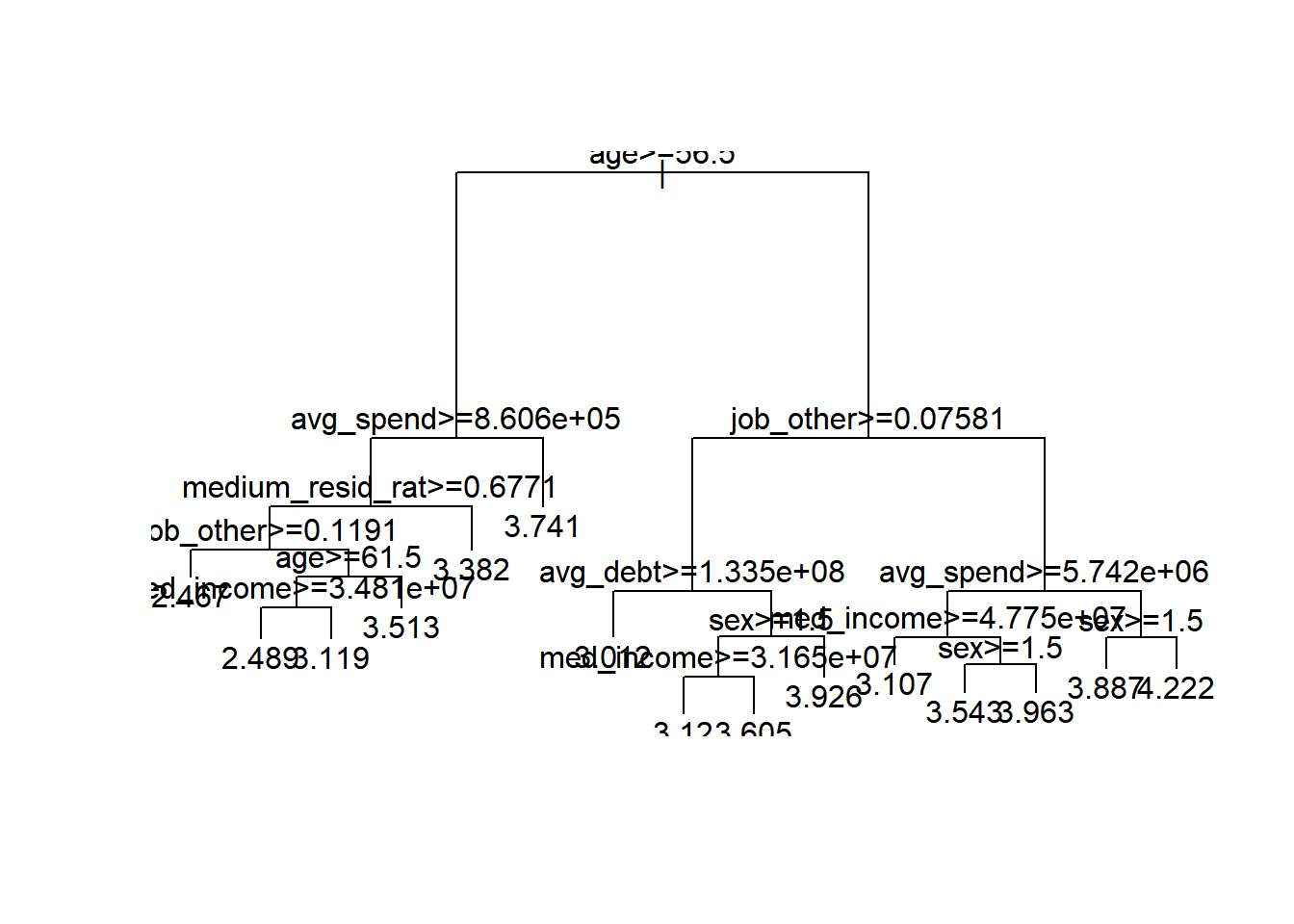
rpart\_test <- rpart(avg\_credit\_rat~. , data = jeju\_data)  
plot(rpart\_test)  
text(rpart\_test)



jeju\_dt\_data <- jeju\_data %>% filter(jeju\_data$medium\_resid\_rat >= 0)  
summary(jeju\_dt\_data)

## zip\_cd year\_month x\_axis y\_axis   
## Min. :63013 Min. :201902 Min. :126.3 Min. :33.22   
## 1st Qu.:63127 1st Qu.:201902 1st Qu.:126.5 1st Qu.:33.48   
## Median :63223 Median :201902 Median :126.5 Median :33.49   
## Mean :63253 Mean :201902 Mean :126.5 Mean :33.46   
## 3rd Qu.:63309 3rd Qu.:201902 3rd Qu.:126.6 3rd Qu.:33.51   
## Max. :63644 Max. :201902 Max. :126.9 Max. :33.54   
## sex age job\_majorc job\_smallc   
## Min. :1.000 Min. :24.0 Min. :0.00000 Min. :0.00000   
## 1st Qu.:1.000 1st Qu.:39.0 1st Qu.:0.00000 1st Qu.:0.07609   
## Median :2.000 Median :54.0 Median :0.01961 Median :0.15217   
## Mean :1.506 Mean :52.9 Mean :0.02772 Mean :0.15522   
## 3rd Qu.:2.000 3rd Qu.:69.0 3rd Qu.:0.04598 3rd Qu.:0.22619   
## Max. :2.000 Max. :99.0 Max. :0.21053 Max. :0.50000   
## job\_public job\_profession job\_self job\_none   
## Min. :0.00000 Min. :0.000000 Min. :0.00000 Min. :0.1481   
## 1st Qu.:0.00000 1st Qu.:0.000000 1st Qu.:0.06061 1st Qu.:0.4000   
## Median :0.02857 Median :0.008772 Median :0.13793 Median :0.5224   
## Mean :0.04128 Mean :0.018161 Mean :0.15138 Mean :0.5547   
## 3rd Qu.:0.06494 3rd Qu.:0.028571 3rd Qu.:0.21778 3rd Qu.:0.7059   
## Max. :0.33333 Max. :0.268293 Max. :0.75410 Max. :1.0000   
## job\_other avg\_income med\_income   
## Min. :0.00000 Min. : 0 Min. : 0   
## 1st Qu.:0.00000 1st Qu.: 26636000 1st Qu.:25630000   
## Median :0.04615 Median : 32257000 Median :30000000   
## Mean :0.05154 Mean : 34779315 Mean :31051446   
## 3rd Qu.:0.08108 3rd Qu.: 40126000 3rd Qu.:35500000   
## Max. :0.29412 Max. :387243000 Max. :81580000   
## avg\_spend avg\_foreign\_spend avg\_debt   
## Min. : 0 Min. : -2581 Min. :0.000e+00   
## 1st Qu.: 2462894 1st Qu.: 5387 1st Qu.:3.214e+07   
## Median : 4483097 Median : 42770 Median :6.166e+07   
## Mean : 4607303 Mean : 72705 Mean :8.080e+07   
## 3rd Qu.: 6142492 3rd Qu.: 95626 3rd Qu.:1.006e+08   
## Max. :28653948 Max. :3187744 Max. :2.061e+09   
## avg\_debt\_credit avg\_debt\_noneb avg\_debt\_mortgage   
## Min. : 0 Min. :0.000e+00 Min. :0.000e+00   
## 1st Qu.: 11869857 1st Qu.:1.279e+07 1st Qu.:4.100e+07   
## Median : 18819242 Median :2.608e+07 Median :8.703e+07   
## Mean : 20389970 Mean :4.061e+07 Mean :9.523e+07   
## 3rd Qu.: 26252619 3rd Qu.:4.733e+07 3rd Qu.:1.316e+08   
## Max. :294313599 Max. :1.283e+09 Max. :1.106e+09   
## avg\_debt\_deposit avg\_debt\_collateral avg\_credit\_rat medium\_resid\_rat  
## Min. : 0 Min. :0.000e+00 Min. :2.000 Min. :0.0000   
## 1st Qu.: 0 1st Qu.:4.872e+07 1st Qu.:3.000 1st Qu.:0.0000   
## Median : 3762910 Median :9.927e+07 Median :4.000 Median :0.5000   
## Mean : 11160590 Mean :1.286e+08 Mean :3.587 Mean :0.5074   
## 3rd Qu.: 11000000 3rd Qu.:1.665e+08 3rd Qu.:4.000 3rd Qu.:1.0000   
## Max. :692740084 Max. :4.083e+09 Max. :6.000 Max. :1.0000   
## large\_resid\_rat vehicle\_own\_rat   
## Min. :0.00000 Min. :0.00000   
## 1st Qu.:0.00000 1st Qu.:0.00000   
## Median :0.00000 Median :0.02830   
## Mean :0.06972 Mean :0.03475   
## 3rd Qu.:0.00000 3rd Qu.:0.05556   
## Max. :1.00000 Max. :0.28571

rpart\_test <- rpart(avg\_credit\_rat~. , data = jeju\_dt\_data)  
  
plot(rpart\_test)  
text(rpart\_test)



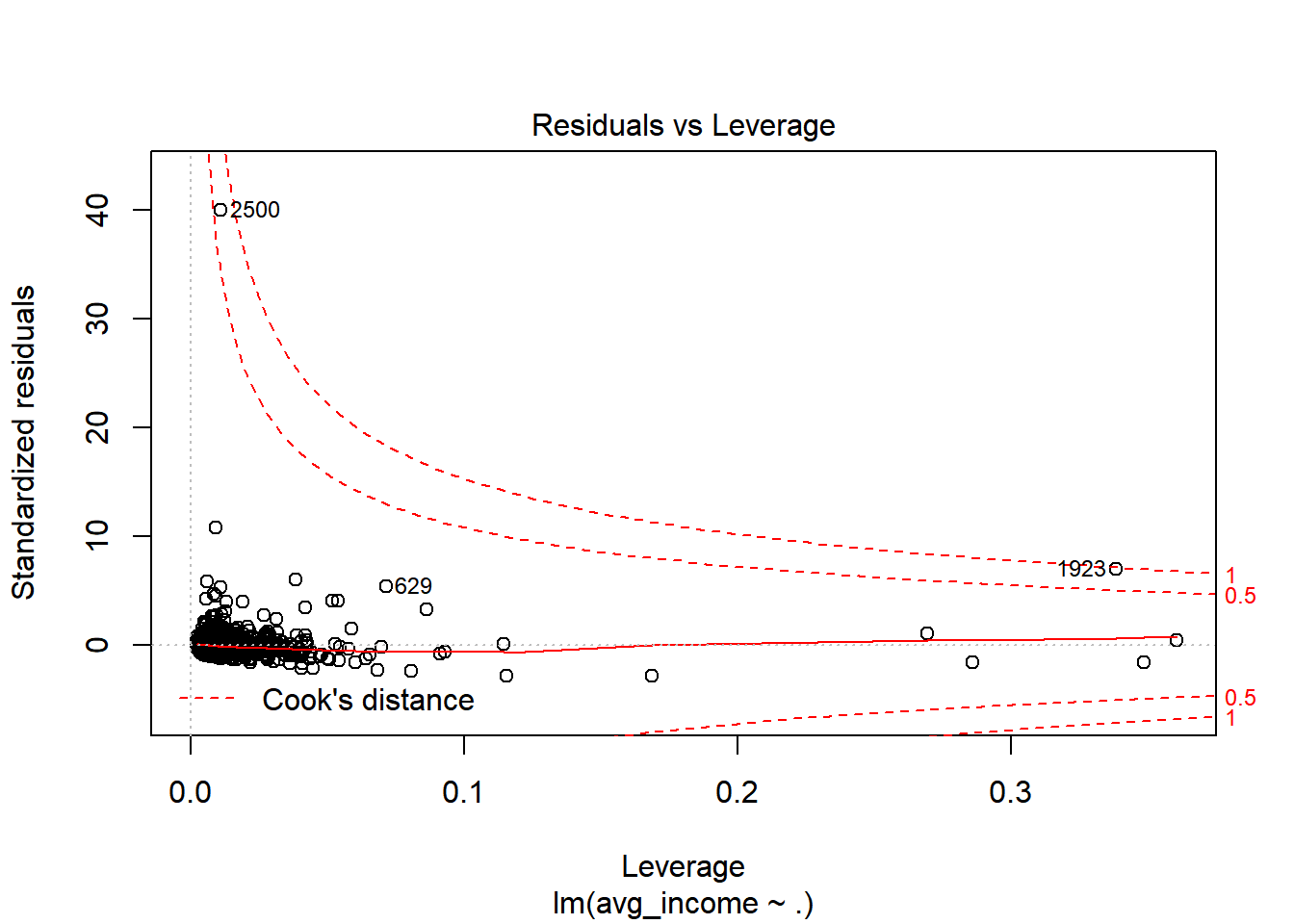
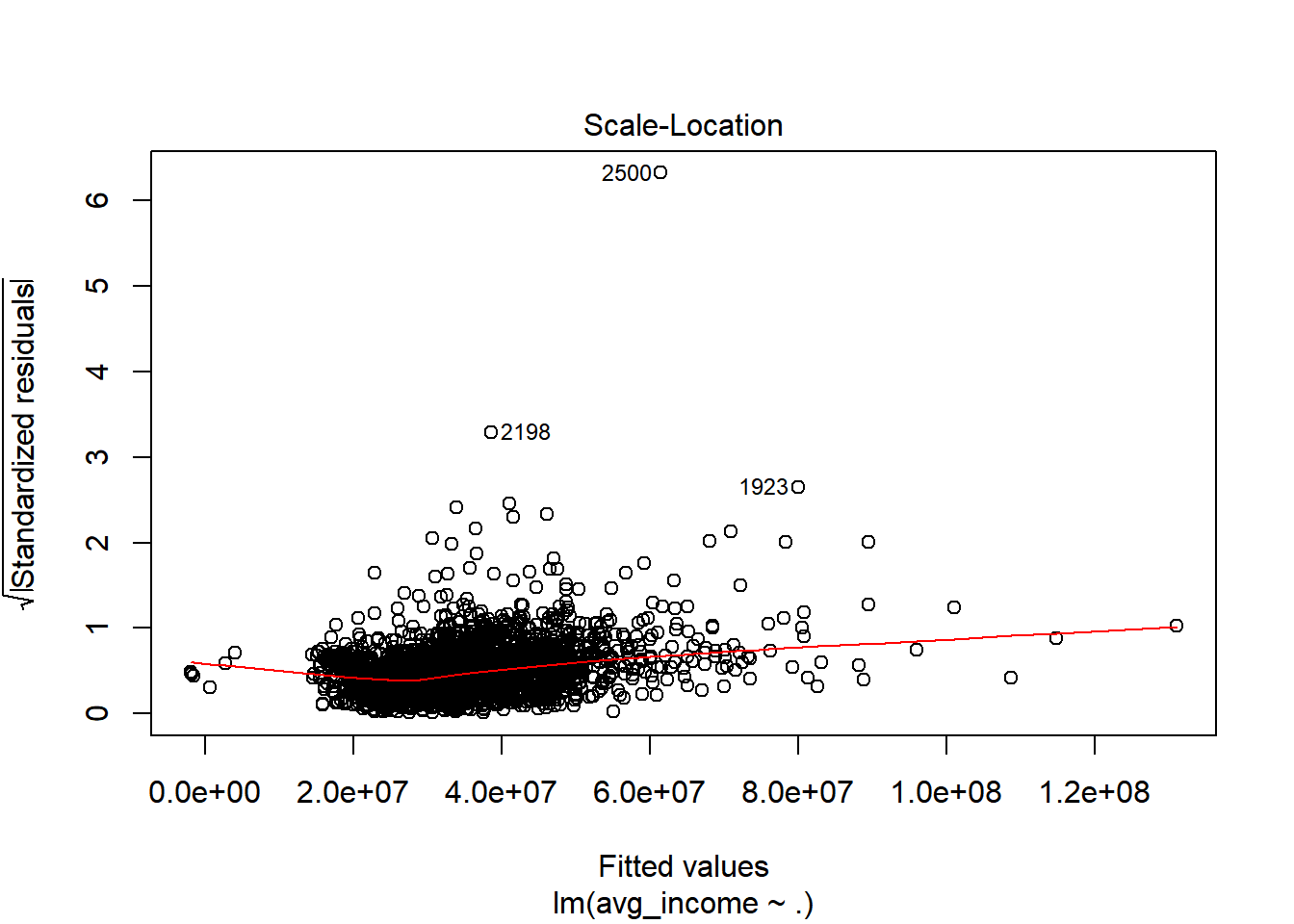
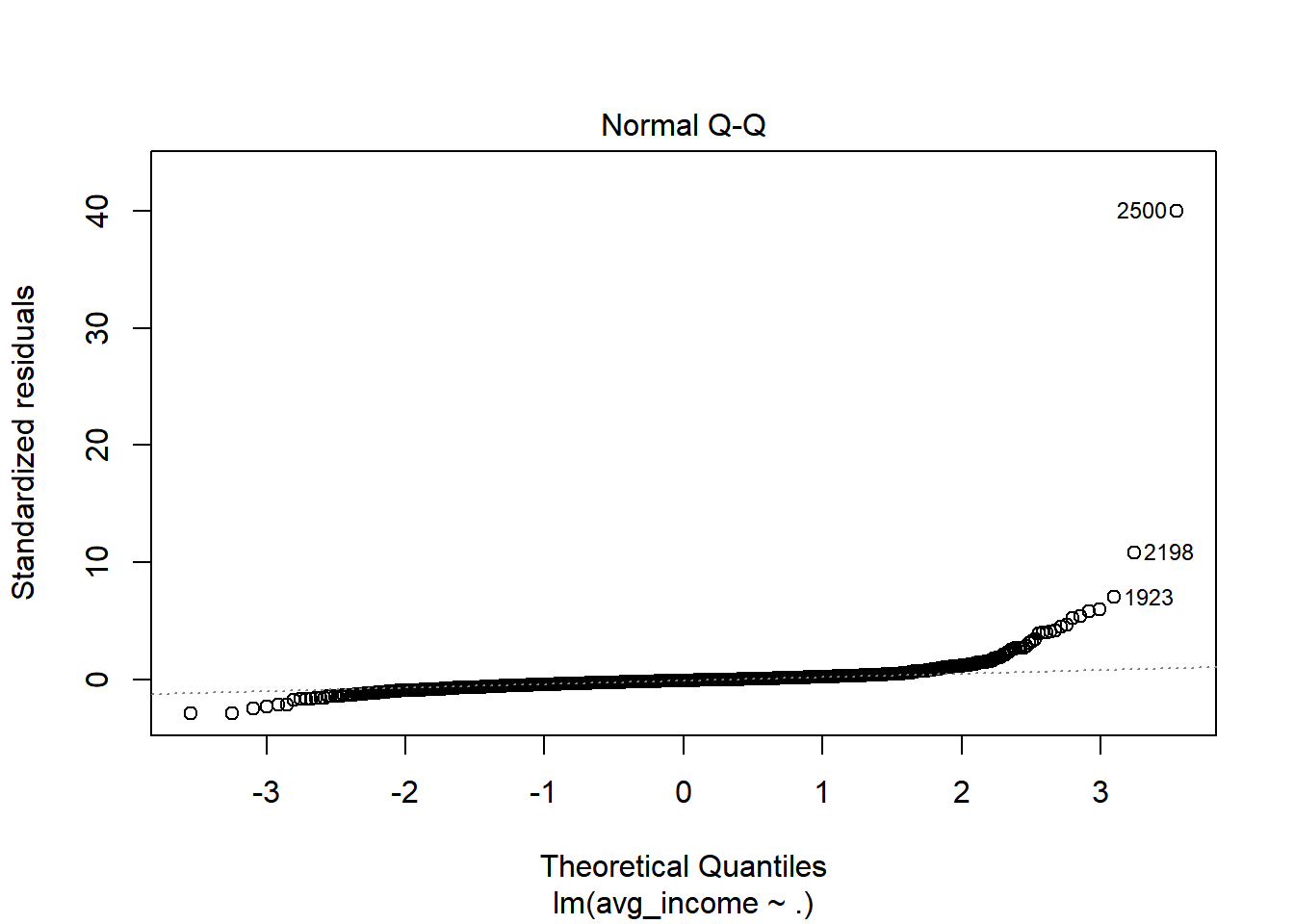
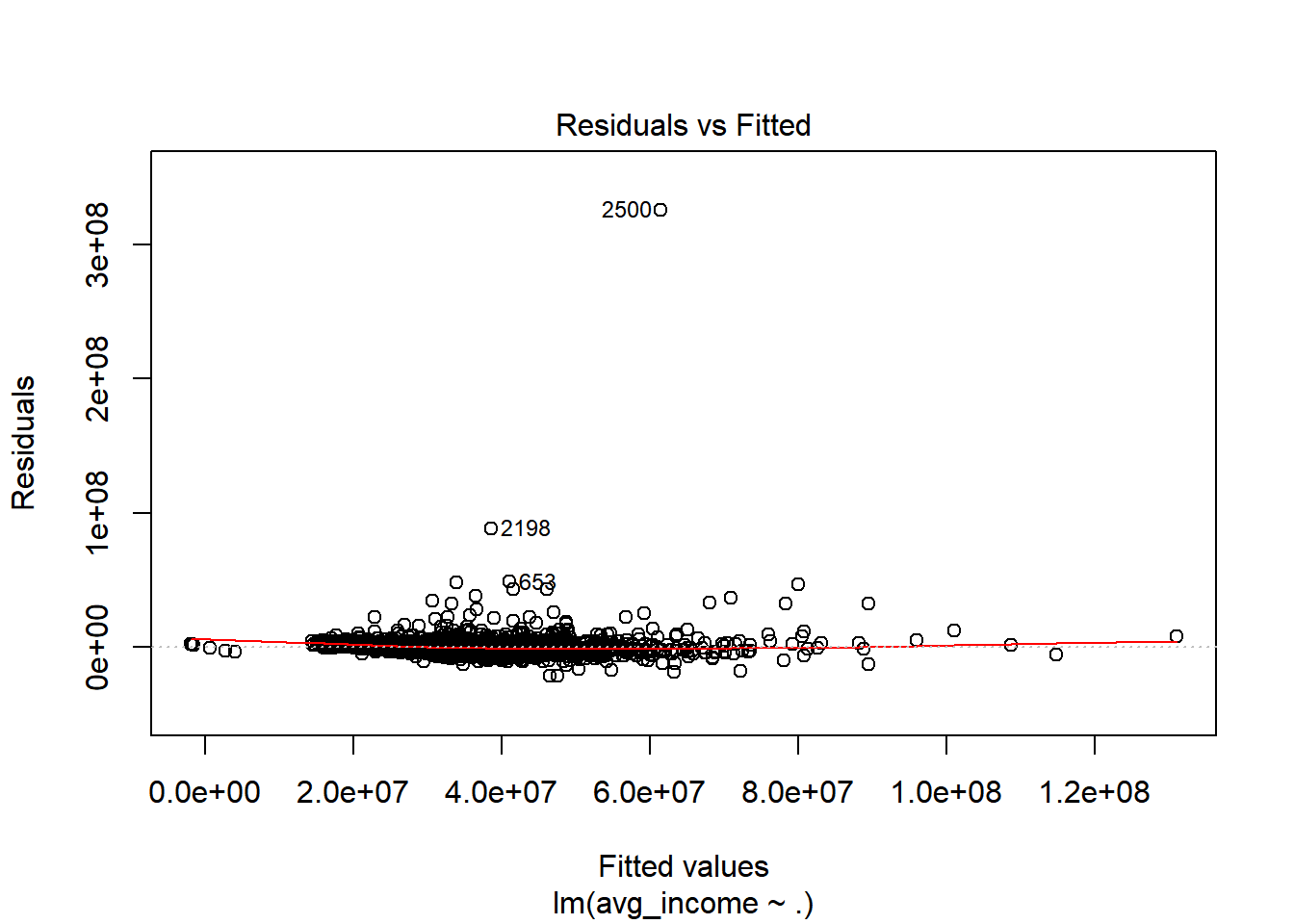
test\_l <-lm(data = jeju\_dt\_data, avg\_income~.)   
test\_l

##   
## Call:  
## lm(formula = avg\_income ~ ., data = jeju\_dt\_data)  
##   
## Coefficients:  
## (Intercept) zip\_cd year\_month   
## -1.907e+11 -3.449e+02 NA   
## x\_axis y\_axis sex   
## -2.249e+05 -3.827e+06 -2.640e+06   
## age job\_majorc job\_smallc   
## 3.545e+04 1.909e+11 1.909e+11   
## job\_public job\_profession job\_self   
## 1.909e+11 1.909e+11 1.909e+11   
## job\_none job\_other med\_income   
## 1.909e+11 1.909e+11 1.050e+00   
## avg\_spend avg\_foreign\_spend avg\_debt   
## 5.316e-01 3.857e+00 2.025e-02   
## avg\_debt\_credit avg\_debt\_noneb avg\_debt\_mortgage   
## 1.708e-02 -9.313e-04 3.062e-04   
## avg\_debt\_deposit avg\_debt\_collateral avg\_credit\_rat   
## 2.410e-02 1.444e-03 -2.452e+05   
## medium\_resid\_rat large\_resid\_rat vehicle\_own\_rat   
## -1.080e+05 2.154e+06 -1.680e+07

summary(test\_l)

##   
## Call:  
## lm(formula = avg\_income ~ ., data = jeju\_dt\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -22013277 -2140336 -369485 1201985 325751129   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.907e+11 2.239e+11 -0.852 0.394325   
## zip\_cd -3.449e+02 2.344e+03 -0.147 0.883003   
## year\_month NA NA NA NA   
## x\_axis -2.249e+05 3.173e+06 -0.071 0.943496   
## y\_axis -3.827e+06 4.089e+06 -0.936 0.349454   
## sex -2.640e+06 4.433e+05 -5.954 2.98e-09 \*\*\*  
## age 3.545e+04 1.343e+04 2.640 0.008335 \*\*   
## job\_majorc 1.909e+11 2.239e+11 0.853 0.393909   
## job\_smallc 1.909e+11 2.239e+11 0.853 0.393893   
## job\_public 1.909e+11 2.239e+11 0.853 0.393891   
## job\_profession 1.909e+11 2.239e+11 0.853 0.393843   
## job\_self 1.909e+11 2.239e+11 0.853 0.393870   
## job\_none 1.909e+11 2.239e+11 0.853 0.393880   
## job\_other 1.909e+11 2.239e+11 0.853 0.393854   
## med\_income 1.050e+00 4.327e-02 24.272 < 2e-16 \*\*\*  
## avg\_spend 5.316e-01 1.046e-01 5.083 3.99e-07 \*\*\*  
## avg\_foreign\_spend 3.857e+00 1.565e+00 2.465 0.013786 \*   
## avg\_debt 2.025e-02 4.853e-03 4.172 3.13e-05 \*\*\*  
## avg\_debt\_credit 1.708e-02 1.561e-02 1.094 0.274014   
## avg\_debt\_noneb -9.313e-04 3.734e-03 -0.249 0.803055   
## avg\_debt\_mortgage 3.062e-04 2.672e-03 0.115 0.908755   
## avg\_debt\_deposit 2.410e-02 6.301e-03 3.824 0.000135 \*\*\*  
## avg\_debt\_collateral 1.444e-03 2.290e-03 0.630 0.528466   
## avg\_credit\_rat -2.452e+05 2.962e+05 -0.828 0.407942   
## medium\_resid\_rat -1.080e+05 4.399e+05 -0.245 0.806120   
## large\_resid\_rat 2.154e+06 9.774e+05 2.204 0.027594 \*   
## vehicle\_own\_rat -1.680e+07 5.927e+06 -2.835 0.004625 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 8189000 on 2523 degrees of freedom  
## Multiple R-squared: 0.6627, Adjusted R-squared: 0.6594   
## F-statistic: 198.3 on 25 and 2523 DF, p-value: < 2.2e-16

plot(test\_l)



data<-iris  
scale.data<-data.frame(lapply(data[,1:4], function(x) scale(x)))  
scale.data

## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## 1 -0.89767388 1.01560199 -1.33575163 -1.3110521482  
## 2 -1.13920048 -0.13153881 -1.33575163 -1.3110521482  
## 3 -1.38072709 0.32731751 -1.39239929 -1.3110521482  
## 4 -1.50149039 0.09788935 -1.27910398 -1.3110521482  
## 5 -1.01843718 1.24503015 -1.33575163 -1.3110521482  
## 6 -0.53538397 1.93331463 -1.16580868 -1.0486667950  
## 7 -1.50149039 0.78617383 -1.33575163 -1.1798594716  
## 8 -1.01843718 0.78617383 -1.27910398 -1.3110521482  
## 9 -1.74301699 -0.36096697 -1.33575163 -1.3110521482  
## 10 -1.13920048 0.09788935 -1.27910398 -1.4422448248  
## 11 -0.53538397 1.47445831 -1.27910398 -1.3110521482  
## 12 -1.25996379 0.78617383 -1.22245633 -1.3110521482  
## 13 -1.25996379 -0.13153881 -1.33575163 -1.4422448248  
## 14 -1.86378030 -0.13153881 -1.50569459 -1.4422448248  
## 15 -0.05233076 2.16274279 -1.44904694 -1.3110521482  
## 16 -0.17309407 3.08045544 -1.27910398 -1.0486667950  
## 17 -0.53538397 1.93331463 -1.39239929 -1.0486667950  
## 18 -0.89767388 1.01560199 -1.33575163 -1.1798594716  
## 19 -0.17309407 1.70388647 -1.16580868 -1.1798594716  
## 20 -0.89767388 1.70388647 -1.27910398 -1.1798594716  
## 21 -0.53538397 0.78617383 -1.16580868 -1.3110521482  
## 22 -0.89767388 1.47445831 -1.27910398 -1.0486667950  
## 23 -1.50149039 1.24503015 -1.56234224 -1.3110521482  
## 24 -0.89767388 0.55674567 -1.16580868 -0.9174741184  
## 25 -1.25996379 0.78617383 -1.05251337 -1.3110521482  
## 26 -1.01843718 -0.13153881 -1.22245633 -1.3110521482  
## 27 -1.01843718 0.78617383 -1.22245633 -1.0486667950  
## 28 -0.77691058 1.01560199 -1.27910398 -1.3110521482  
## 29 -0.77691058 0.78617383 -1.33575163 -1.3110521482  
## 30 -1.38072709 0.32731751 -1.22245633 -1.3110521482  
## 31 -1.25996379 0.09788935 -1.22245633 -1.3110521482  
## 32 -0.53538397 0.78617383 -1.27910398 -1.0486667950  
## 33 -0.77691058 2.39217095 -1.27910398 -1.4422448248  
## 34 -0.41462067 2.62159911 -1.33575163 -1.3110521482  
## 35 -1.13920048 0.09788935 -1.27910398 -1.3110521482  
## 36 -1.01843718 0.32731751 -1.44904694 -1.3110521482  
## 37 -0.41462067 1.01560199 -1.39239929 -1.3110521482  
## 38 -1.13920048 1.24503015 -1.33575163 -1.4422448248  
## 39 -1.74301699 -0.13153881 -1.39239929 -1.3110521482  
## 40 -0.89767388 0.78617383 -1.27910398 -1.3110521482  
## 41 -1.01843718 1.01560199 -1.39239929 -1.1798594716  
## 42 -1.62225369 -1.73753594 -1.39239929 -1.1798594716  
## 43 -1.74301699 0.32731751 -1.39239929 -1.3110521482  
## 44 -1.01843718 1.01560199 -1.22245633 -0.7862814418  
## 45 -0.89767388 1.70388647 -1.05251337 -1.0486667950  
## 46 -1.25996379 -0.13153881 -1.33575163 -1.1798594716  
## 47 -0.89767388 1.70388647 -1.22245633 -1.3110521482  
## 48 -1.50149039 0.32731751 -1.33575163 -1.3110521482  
## 49 -0.65614727 1.47445831 -1.27910398 -1.3110521482  
## 50 -1.01843718 0.55674567 -1.33575163 -1.3110521482  
## 51 1.39682886 0.32731751 0.53362088 0.2632599711  
## 52 0.67224905 0.32731751 0.42032558 0.3944526477  
## 53 1.27606556 0.09788935 0.64691619 0.3944526477  
## 54 -0.41462067 -1.73753594 0.13708732 0.1320672944  
## 55 0.79301235 -0.59039513 0.47697323 0.3944526477  
## 56 -0.17309407 -0.59039513 0.42032558 0.1320672944  
## 57 0.55148575 0.55674567 0.53362088 0.5256453243  
## 58 -1.13920048 -1.50810778 -0.25944625 -0.2615107354  
## 59 0.91377565 -0.36096697 0.47697323 0.1320672944  
## 60 -0.77691058 -0.81982329 0.08043967 0.2632599711  
## 61 -1.01843718 -2.42582042 -0.14615094 -0.2615107354  
## 62 0.06843254 -0.13153881 0.25038262 0.3944526477  
## 63 0.18919584 -1.96696410 0.13708732 -0.2615107354  
## 64 0.30995914 -0.36096697 0.53362088 0.2632599711  
## 65 -0.29385737 -0.36096697 -0.08950329 0.1320672944  
## 66 1.03453895 0.09788935 0.36367793 0.2632599711  
## 67 -0.29385737 -0.13153881 0.42032558 0.3944526477  
## 68 -0.05233076 -0.81982329 0.19373497 -0.2615107354  
## 69 0.43072244 -1.96696410 0.42032558 0.3944526477  
## 70 -0.29385737 -1.27867961 0.08043967 -0.1303180588  
## 71 0.06843254 0.32731751 0.59026853 0.7880306775  
## 72 0.30995914 -0.59039513 0.13708732 0.1320672944  
## 73 0.55148575 -1.27867961 0.64691619 0.3944526477  
## 74 0.30995914 -0.59039513 0.53362088 0.0008746178  
## 75 0.67224905 -0.36096697 0.30703027 0.1320672944  
## 76 0.91377565 -0.13153881 0.36367793 0.2632599711  
## 77 1.15530226 -0.59039513 0.59026853 0.2632599711  
## 78 1.03453895 -0.13153881 0.70356384 0.6568380009  
## 79 0.18919584 -0.36096697 0.42032558 0.3944526477  
## 80 -0.17309407 -1.04925145 -0.14615094 -0.2615107354  
## 81 -0.41462067 -1.50810778 0.02379201 -0.1303180588  
## 82 -0.41462067 -1.50810778 -0.03285564 -0.2615107354  
## 83 -0.05233076 -0.81982329 0.08043967 0.0008746178  
## 84 0.18919584 -0.81982329 0.76021149 0.5256453243  
## 85 -0.53538397 -0.13153881 0.42032558 0.3944526477  
## 86 0.18919584 0.78617383 0.42032558 0.5256453243  
## 87 1.03453895 0.09788935 0.53362088 0.3944526477  
## 88 0.55148575 -1.73753594 0.36367793 0.1320672944  
## 89 -0.29385737 -0.13153881 0.19373497 0.1320672944  
## 90 -0.41462067 -1.27867961 0.13708732 0.1320672944  
## 91 -0.41462067 -1.04925145 0.36367793 0.0008746178  
## 92 0.30995914 -0.13153881 0.47697323 0.2632599711  
## 93 -0.05233076 -1.04925145 0.13708732 0.0008746178  
## 94 -1.01843718 -1.73753594 -0.25944625 -0.2615107354  
## 95 -0.29385737 -0.81982329 0.25038262 0.1320672944  
## 96 -0.17309407 -0.13153881 0.25038262 0.0008746178  
## 97 -0.17309407 -0.36096697 0.25038262 0.1320672944  
## 98 0.43072244 -0.36096697 0.30703027 0.1320672944  
## 99 -0.89767388 -1.27867961 -0.42938920 -0.1303180588  
## 100 -0.17309407 -0.59039513 0.19373497 0.1320672944  
## 101 0.55148575 0.55674567 1.27004036 1.7063794137  
## 102 -0.05233076 -0.81982329 0.76021149 0.9192233541  
## 103 1.51759216 -0.13153881 1.21339271 1.1816087073  
## 104 0.55148575 -0.36096697 1.04344975 0.7880306775  
## 105 0.79301235 -0.13153881 1.15674505 1.3128013839  
## 106 2.12140867 -0.13153881 1.60992627 1.1816087073  
## 107 -1.13920048 -1.27867961 0.42032558 0.6568380009  
## 108 1.75911877 -0.36096697 1.43998331 0.7880306775  
## 109 1.03453895 -1.27867961 1.15674505 0.7880306775  
## 110 1.63835547 1.24503015 1.32668801 1.7063794137  
## 111 0.79301235 0.32731751 0.76021149 1.0504160307  
## 112 0.67224905 -0.81982329 0.87350679 0.9192233541  
## 113 1.15530226 -0.13153881 0.98680210 1.1816087073  
## 114 -0.17309407 -1.27867961 0.70356384 1.0504160307  
## 115 -0.05233076 -0.59039513 0.76021149 1.5751867371  
## 116 0.67224905 0.32731751 0.87350679 1.4439940605  
## 117 0.79301235 -0.13153881 0.98680210 0.7880306775  
## 118 2.24217198 1.70388647 1.66657392 1.3128013839  
## 119 2.24217198 -1.04925145 1.77986923 1.4439940605  
## 120 0.18919584 -1.96696410 0.70356384 0.3944526477  
## 121 1.27606556 0.32731751 1.10009740 1.4439940605  
## 122 -0.29385737 -0.59039513 0.64691619 1.0504160307  
## 123 2.24217198 -0.59039513 1.66657392 1.0504160307  
## 124 0.55148575 -0.81982329 0.64691619 0.7880306775  
## 125 1.03453895 0.55674567 1.10009740 1.1816087073  
## 126 1.63835547 0.32731751 1.27004036 0.7880306775  
## 127 0.43072244 -0.59039513 0.59026853 0.7880306775  
## 128 0.30995914 -0.13153881 0.64691619 0.7880306775  
## 129 0.67224905 -0.59039513 1.04344975 1.1816087073  
## 130 1.63835547 -0.13153881 1.15674505 0.5256453243  
## 131 1.87988207 -0.59039513 1.32668801 0.9192233541  
## 132 2.48369858 1.70388647 1.49663097 1.0504160307  
## 133 0.67224905 -0.59039513 1.04344975 1.3128013839  
## 134 0.55148575 -0.59039513 0.76021149 0.3944526477  
## 135 0.30995914 -1.04925145 1.04344975 0.2632599711  
## 136 2.24217198 -0.13153881 1.32668801 1.4439940605  
## 137 0.55148575 0.78617383 1.04344975 1.5751867371  
## 138 0.67224905 0.09788935 0.98680210 0.7880306775  
## 139 0.18919584 -0.13153881 0.59026853 0.7880306775  
## 140 1.27606556 0.09788935 0.93015445 1.1816087073  
## 141 1.03453895 0.09788935 1.04344975 1.5751867371  
## 142 1.27606556 0.09788935 0.76021149 1.4439940605  
## 143 -0.05233076 -0.81982329 0.76021149 0.9192233541  
## 144 1.15530226 0.32731751 1.21339271 1.4439940605  
## 145 1.03453895 0.55674567 1.10009740 1.7063794137  
## 146 1.03453895 -0.13153881 0.81685914 1.4439940605  
## 147 0.55148575 -1.27867961 0.70356384 0.9192233541  
## 148 0.79301235 -0.13153881 0.81685914 1.0504160307  
## 149 0.43072244 0.78617383 0.93015445 1.4439940605  
## 150 0.06843254 -0.13153881 0.76021149 0.7880306775

scale.data$Species<-data$Species  
index<-sample(1:nrow(scale.data),round(0.75\*nrow(scale.data)),replace=FALSE)  
clust.train<-scale.data[index,]  
clust.test<-scale.data[-index,]  
  
  
############################  
## 2019.06.03  
##################  
  
  
  
test1<-lm(avg\_credit\_rat~medium\_resid\_rat, data = jeju\_data)  
wholetest11 <- cor.test(jeju\_data$avg\_spend, jeju\_data$avg\_credit\_rat)  
wholetest11

##   
## Pearson's product-moment correlation  
##   
## data: jeju\_data$avg\_spend and jeju\_data$avg\_credit\_rat  
## t = -1.5309, df = 10420, p-value = 0.1258  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.03418524 0.00420440  
## sample estimates:  
## cor   
## -0.01499595

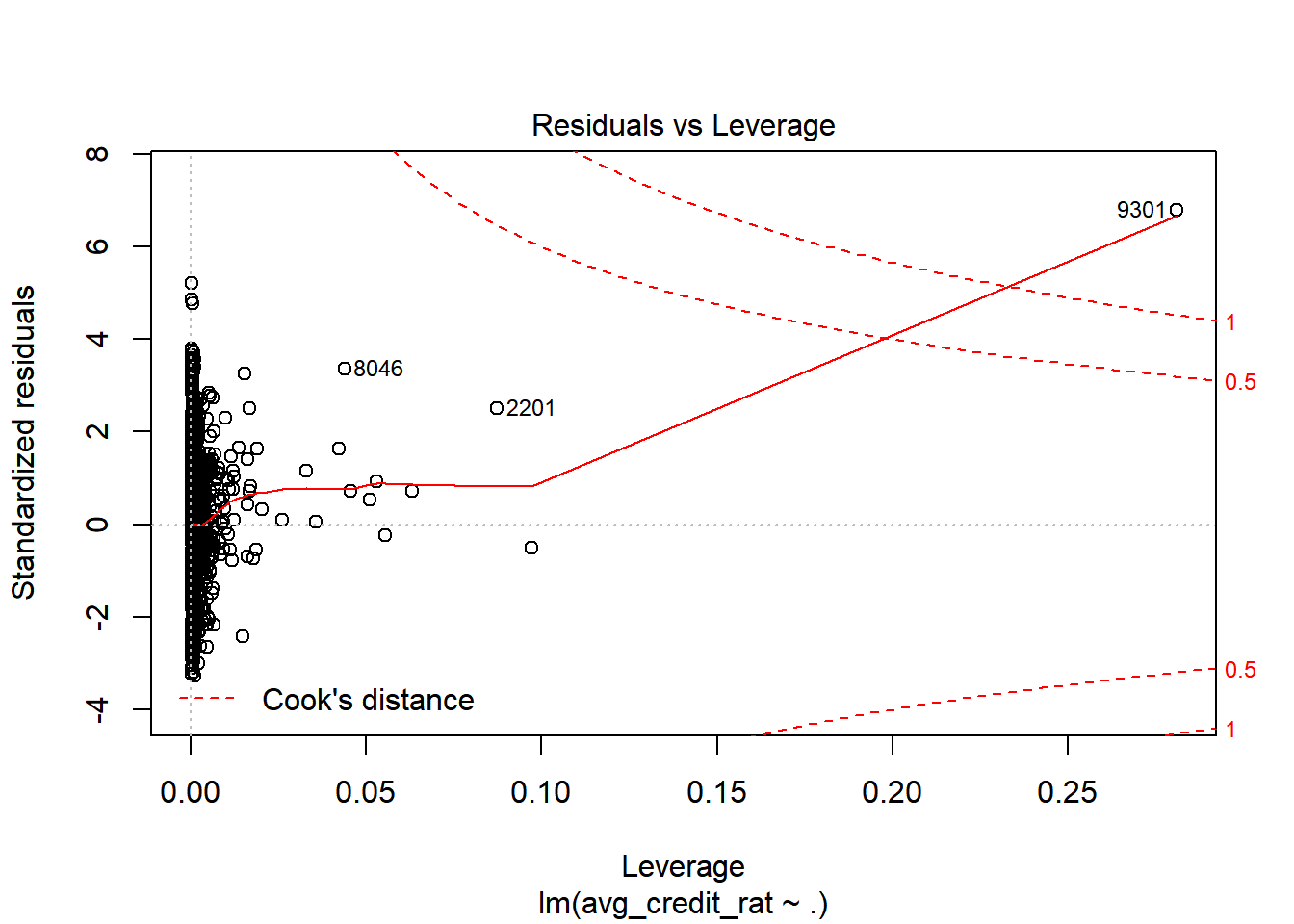
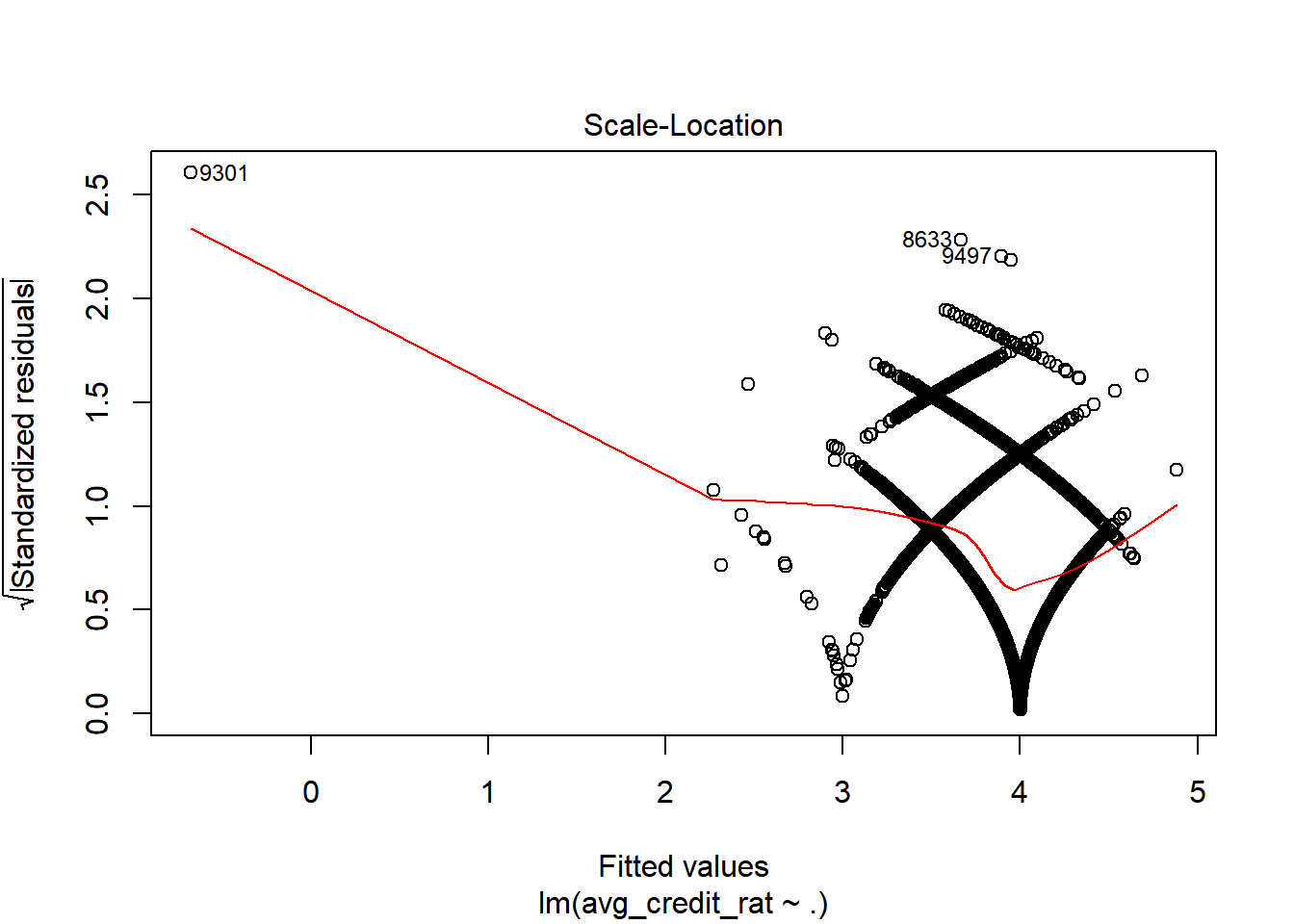
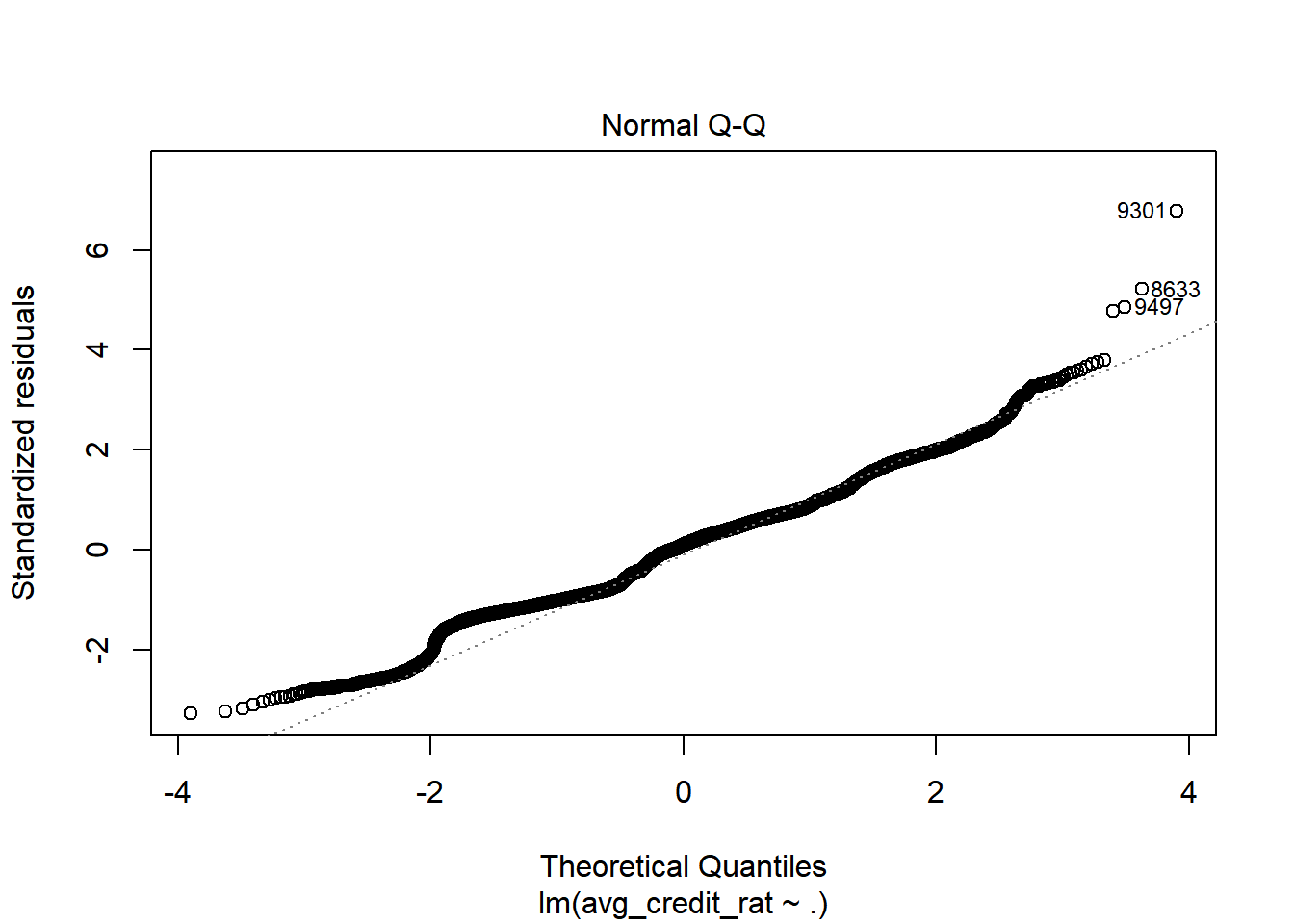
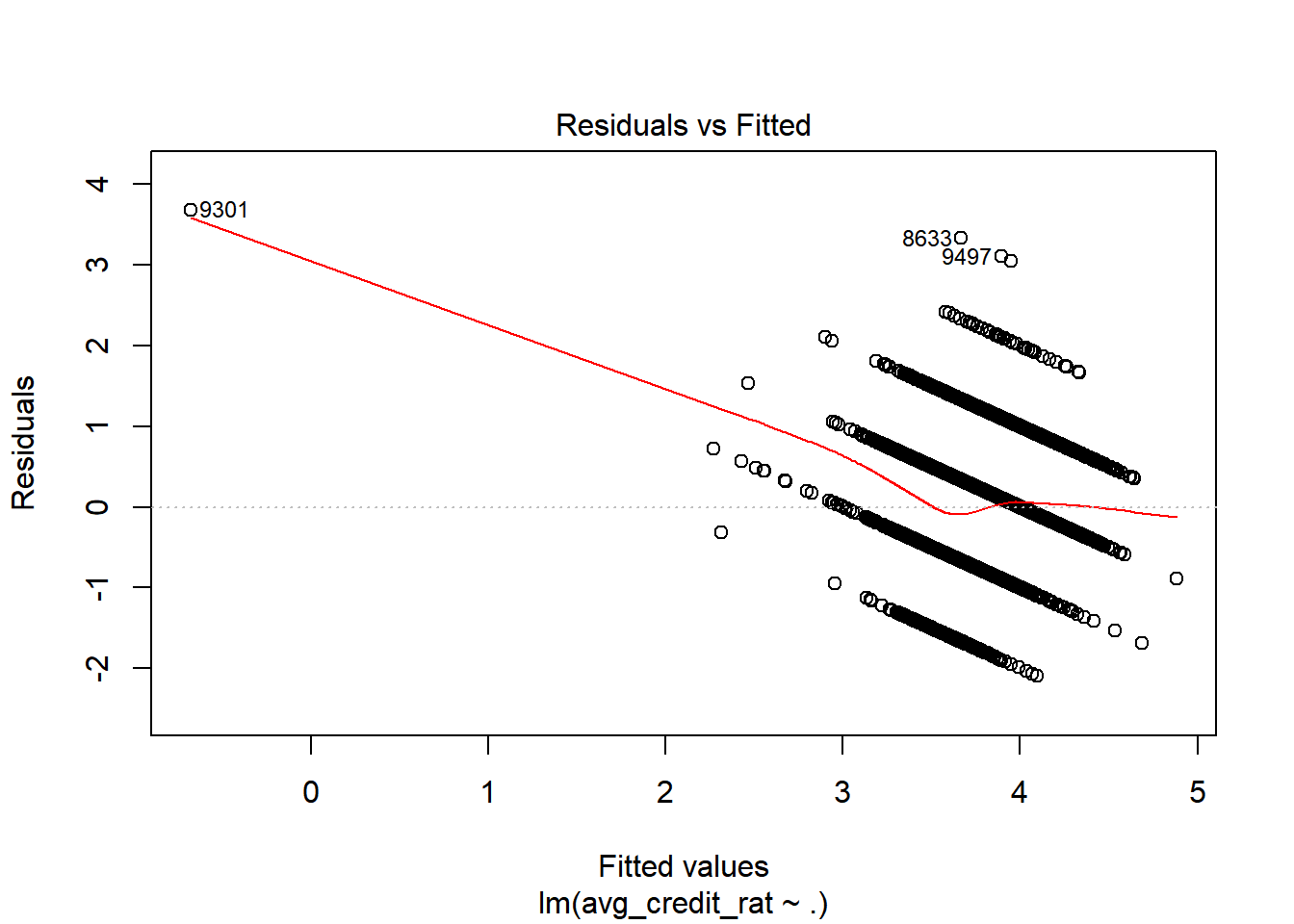
##decision tree###  
  
set.seed(2000) #reproducability setting  
intrain<-caret::createDataPartition(y=jeju\_data$avg\_credit\_rat, p=0.7, list=FALSE)  
  
train<-jeju\_data[intrain, ]  
test<-jeju\_data[-intrain, ]  
  
  
  
  
  
#ptree<-prune(rpartmod, cp= rpartmod$cptable[which.min(rpartmod$cptable[,"xerror"]),"CP"])  
#plot(ptree)  
#text(ptree)  
  
#rpartpred<-predict(ptree, test, type='class')  
#confusionMatrix(rpartpred, table(test$avg\_credit\_rat))  
  
  
# ### party decision tree ###  
# jeju\_data\_party <- jeju\_data[,c(5,6,14,16,18,19,24)]  
# partymod<-ctree(avg\_credit\_rat~., data=jeju\_data\_party)  
# plot(partymod)  
#   
#   
# partymod<-ctree(avg\_credit\_rat~., data=jeju\_data\_party)  
# plot(partymod)  
#   
# partypred<-predict(partymod, test)  
#confusionMatrix(partypred, jeju\_data\_party$avg\_credit\_rat)  
#   
#   
# set.seed(2549) #reproducability setting  
# intrain<-createDataPartition(y = jeju\_data\_party$avg\_credit\_rat, p=0.7, list=FALSE)   
# train<-jeju\_data\_party[intrain, ]  
# test<-jeju\_data\_party[-intrain, ]  
# A <- ctree\_control(maxdepth=20)  
# B <- ctree(avg\_credit\_rat ~ . , data = jeju\_data\_party, controls = A)  
# plot(B, compress=TRUE)  
  
  
###############################  
  
jeju\_data\_second <- jeju\_data[,c(5,6,14,16,18,19,24,27)]  
#jeju\_data\_second  
second\_rm <- lm(data=jeju\_data\_second,avg\_credit\_rat ~.)  
second\_rm

##   
## Call:  
## lm(formula = avg\_credit\_rat ~ ., data = jeju\_data\_second)  
##   
## Coefficients:  
## (Intercept) sex age avg\_income   
## 4.950e+00 -2.625e-01 -9.764e-03 -3.524e-09   
## avg\_spend avg\_debt avg\_debt\_credit vehicle\_own\_rat   
## -3.174e-08 -1.694e-10 -4.227e-09 2.203e+00

summary(second\_rm)

##   
## Call:  
## lm(formula = avg\_credit\_rat ~ ., data = jeju\_data\_second)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.0978 -0.5417 0.0594 0.4124 3.6803   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.950e+00 3.774e-02 131.136 < 2e-16 \*\*\*  
## sex -2.625e-01 1.378e-02 -19.053 < 2e-16 \*\*\*  
## age -9.764e-03 3.616e-04 -27.002 < 2e-16 \*\*\*  
## avg\_income -3.524e-09 5.873e-10 -6.001 2.03e-09 \*\*\*  
## avg\_spend -3.174e-08 2.572e-09 -12.343 < 2e-16 \*\*\*  
## avg\_debt -1.694e-10 7.062e-11 -2.399 0.0164 \*   
## avg\_debt\_credit -4.227e-09 5.473e-10 -7.724 1.23e-14 \*\*\*  
## vehicle\_own\_rat 2.203e+00 1.528e-01 14.419 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.639 on 10414 degrees of freedom  
## Multiple R-squared: 0.15, Adjusted R-squared: 0.1494   
## F-statistic: 262.5 on 7 and 10414 DF, p-value: < 2.2e-16

plot(second\_rm)



#abline(out,col="red")  
  
  
  
####  
  
test\_second<-lm(avg\_credit\_rat~., data = jeju\_data\_second)  
se\_wt <- cor.test(jeju\_data\_second$avg\_income, jeju\_data$avg\_credit\_rat)  
se\_wt

##   
## Pearson's product-moment correlation  
##   
## data: jeju\_data\_second$avg\_income and jeju\_data$avg\_credit\_rat  
## t = -6.5337, df = 10420, p-value = 6.715e-11  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.08297349 -0.04473183  
## sample estimates:  
## cor   
## -0.06387611

sampletest1 <- cor.test(jeju\_data\_second$avg\_spend, jeju\_data\_second$avg\_credit\_rat)  
sampletest1

##   
## Pearson's product-moment correlation  
##   
## data: jeju\_data\_second$avg\_spend and jeju\_data\_second$avg\_credit\_rat  
## t = -1.5309, df = 10420, p-value = 0.1258  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.03418524 0.00420440  
## sample estimates:  
## cor   
## -0.01499595

set.seed(1)  
inTrain <- createDataPartition(y=jeju\_data\_second$avg\_credit\_rat, p=0.7, list=FALSE)  
cb.train <- jeju\_data\_second[inTrain,]  
cb.test <- jeju\_data\_second[-inTrain,]

# FinalProject\_Report2

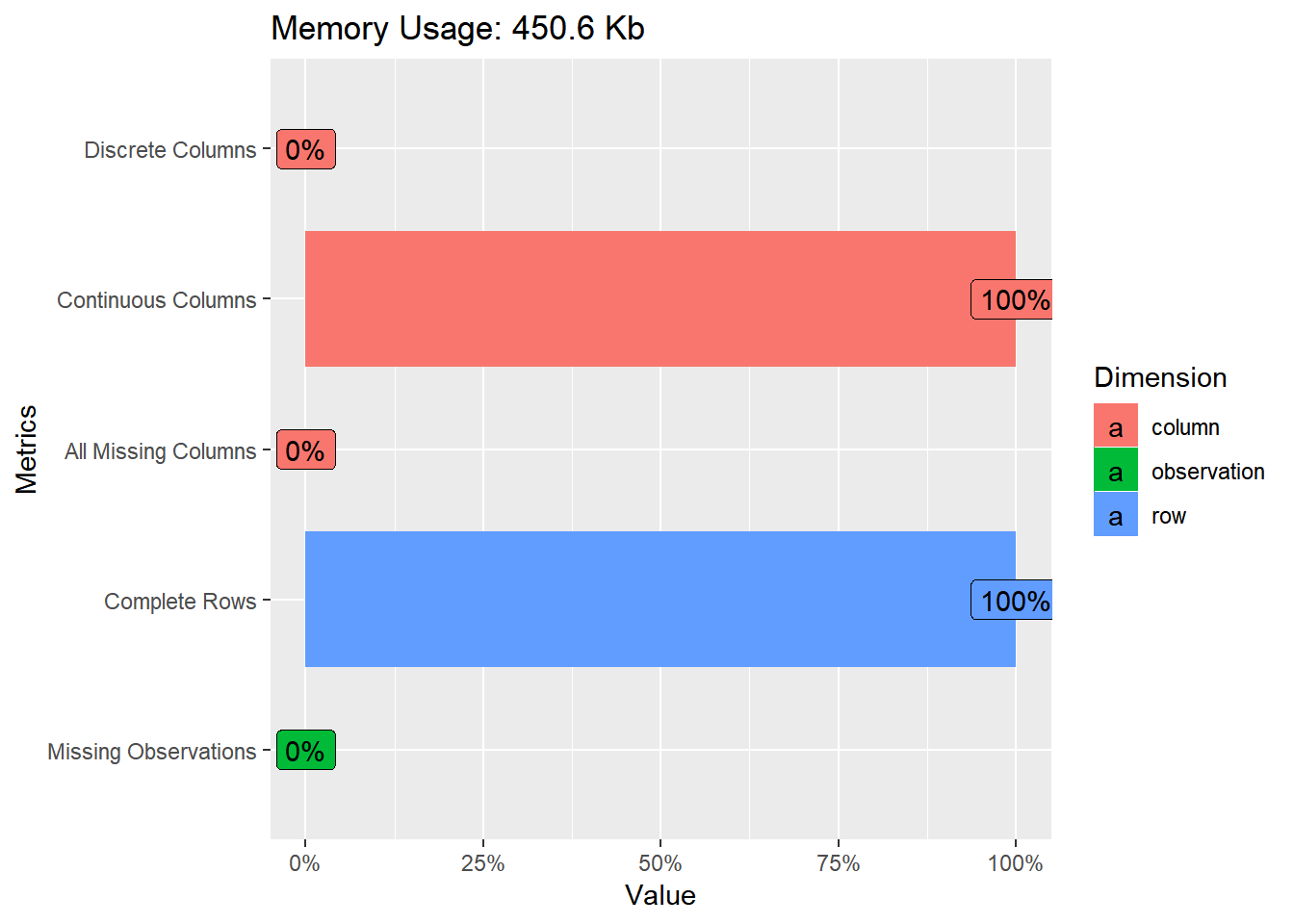
## Shin

## 2019 6 24

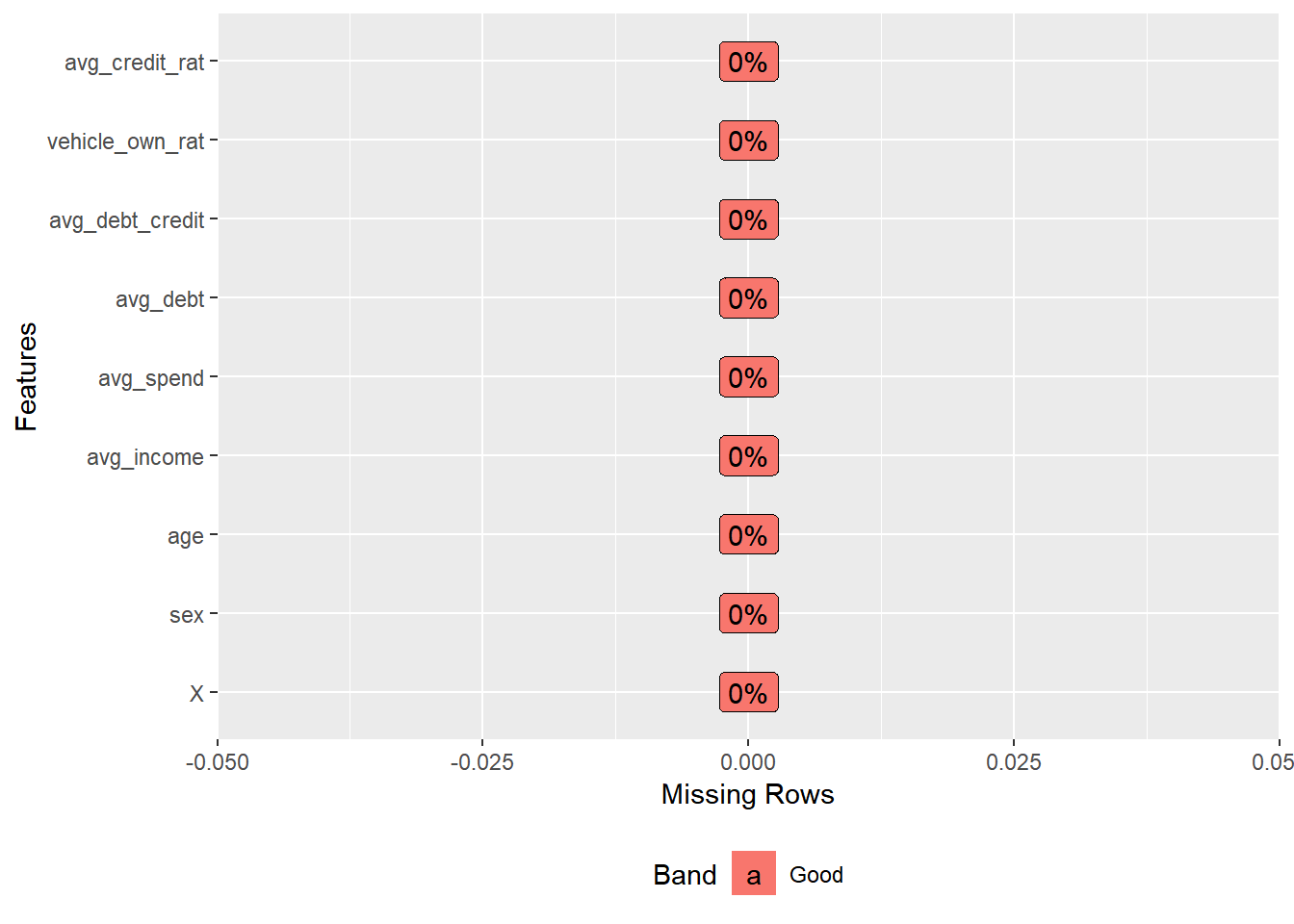
jeju\_data\_party <- read.csv(file = "C:/Users/daily/Desktop/파이널프로젝트\_in R/jeju.csv")  
  
# avg\_credit\_rat <- jeju\_data$avg\_credit\_rat  
# vehicle\_own\_rat <- jeju\_data$vehicle\_own\_rat  
#   
# jeju\_data\_party2 <- cbind(jeju\_data\_party, vehicle\_own\_rat)  
# jeju\_data\_party2 <- cbind(jeju\_data\_party2, avg\_credit\_rat)  
# jeju\_data\_party2 <- jeju\_data\_party2[,-c(7)]  
# jeju\_data\_party2  
#   
# jeju\_data\_party <- jeju\_data\_party2  
#   
# jeju\_data\_party  
# jeju\_data\_b <- jeju\_data\_party  
# jeju\_data\_b  
# jeju\_data\_b <- jeju\_data\_b[,-c(8)]  
  
df <- jeju\_data\_party  
  
# set.seed(10422) #reproducability setting  
# intrain<-createDataPartition(y=df$avg\_credit\_rat, p=0.7, list=FALSE)   
# train<-df[intrain, ]  
# test<-df[-intrain, ]  
#   
# library(rpart)  
# rpartmod<-rpart(avg\_credit\_rat~. , data=train, method="class")  
# plot(rpartmod)  
# text(rpartmod)  
# printcp(rpartmod)  
# plotcp(rpartmod)  
#   
# ptree<-prune(rpartmod, cp= rpartmod$cptable[which.min(rpartmod$cptable[,"xerror"]),"CP"])  
# plot(ptree)  
# text(ptree)  
# rpartpred<-predict(ptree, test, type='class')  
# confusionMatrix(rpartpred,test$avg\_credit\_rat)  
# rpartpred  
# test  
#   
#   
# ### random forest ###  
# library(MASS)   
# library(randomForest)   
# library(caret)  
#   
# set.seed(10000)  
# rf.fit = randomForest(avg\_credit\_rat ~ .  
# , data=jeju\_data\_party2, mtry = floor(sqrt(7)), ntree = 500, importance = T)  
#   
# rf.fit  
#   
  
  
library(DataExplorer)  
introduce(jeju\_data\_party)

## rows columns discrete\_columns continuous\_columns all\_missing\_columns  
## 1 10422 9 0 9 0  
## total\_missing\_values complete\_rows total\_observations memory\_usage  
## 1 0 10422 93798 461456

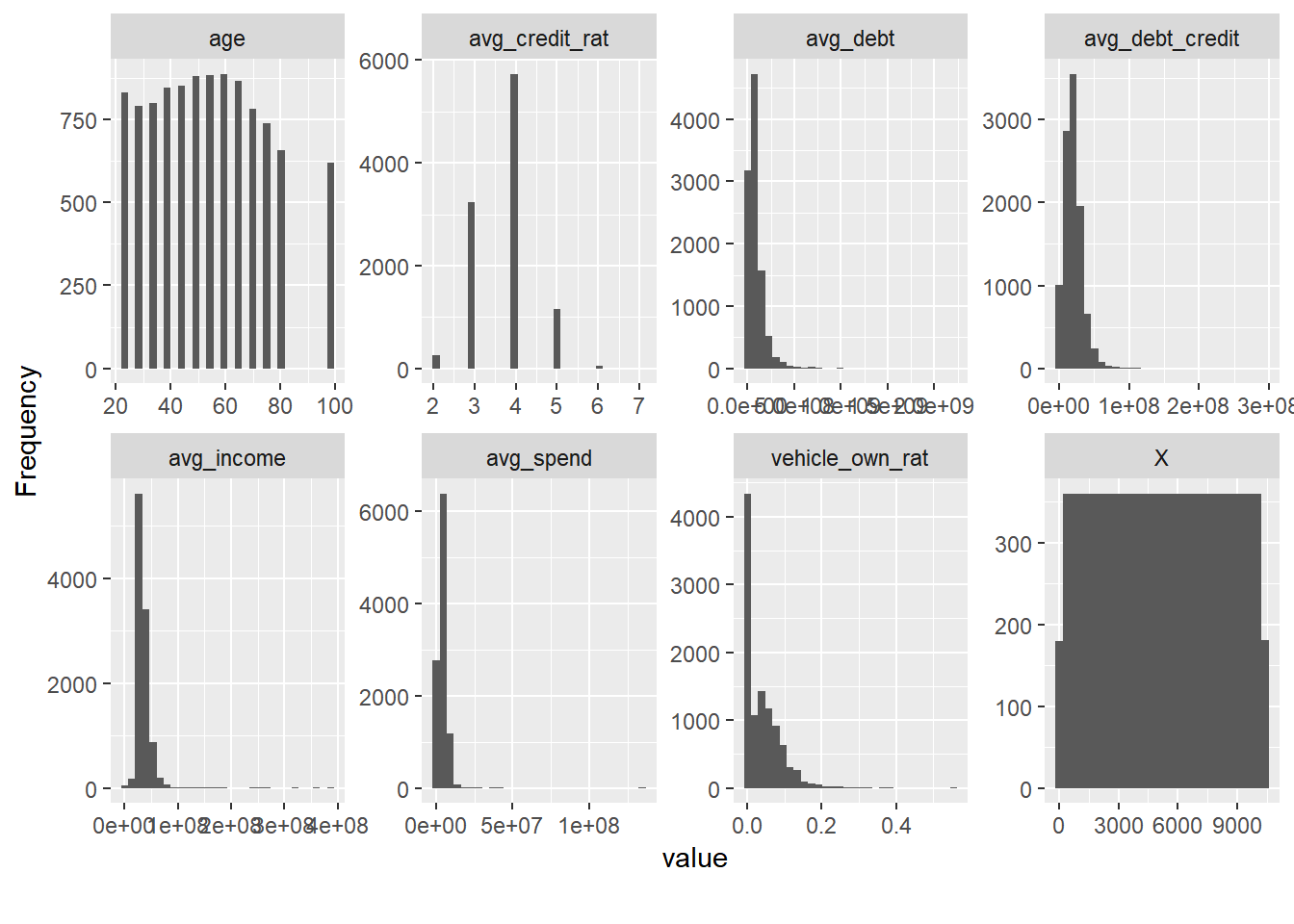
plot\_intro(jeju\_data\_party)



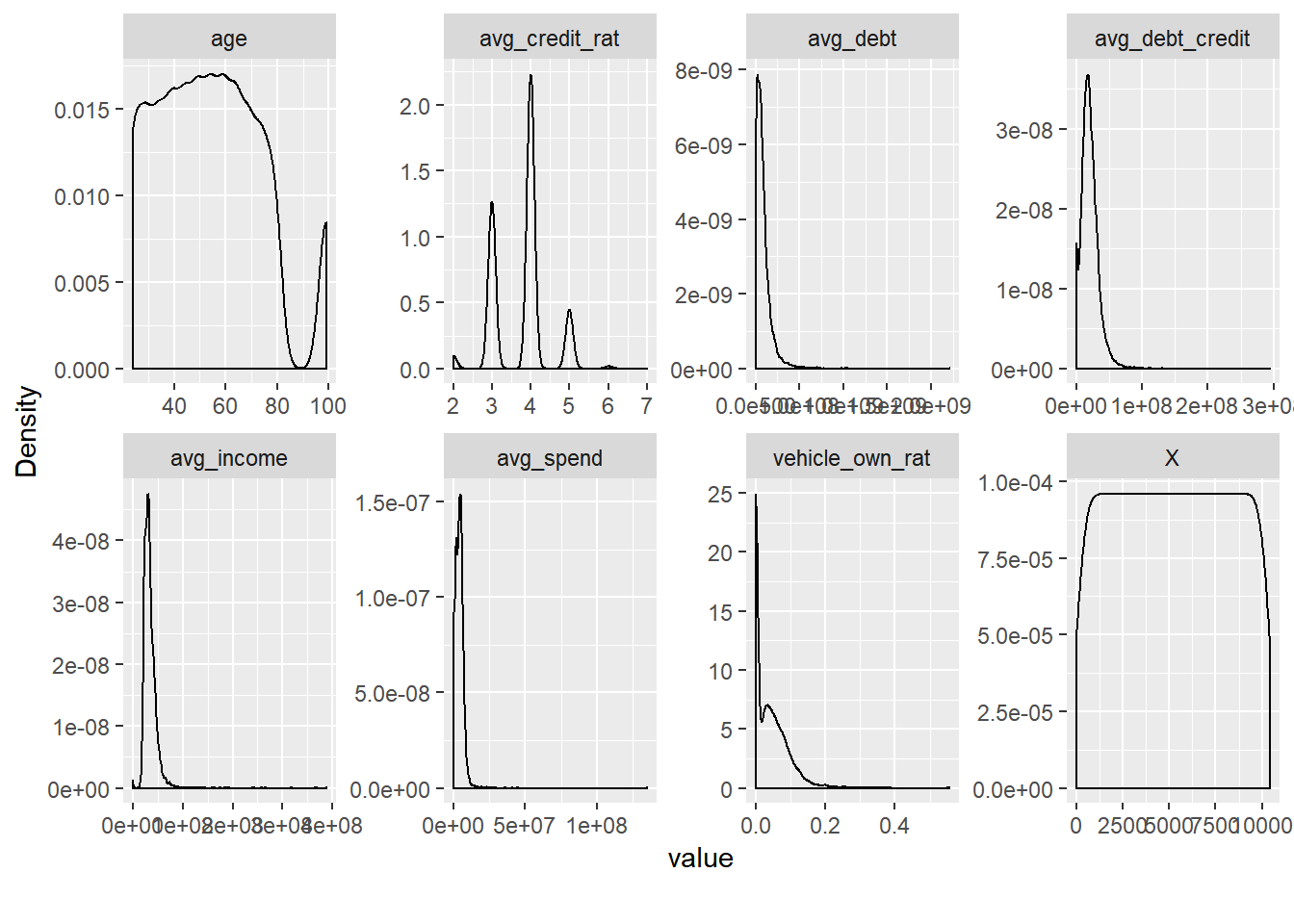
plot\_str(jeju\_data\_party)  
plot\_missing(jeju\_data\_party)



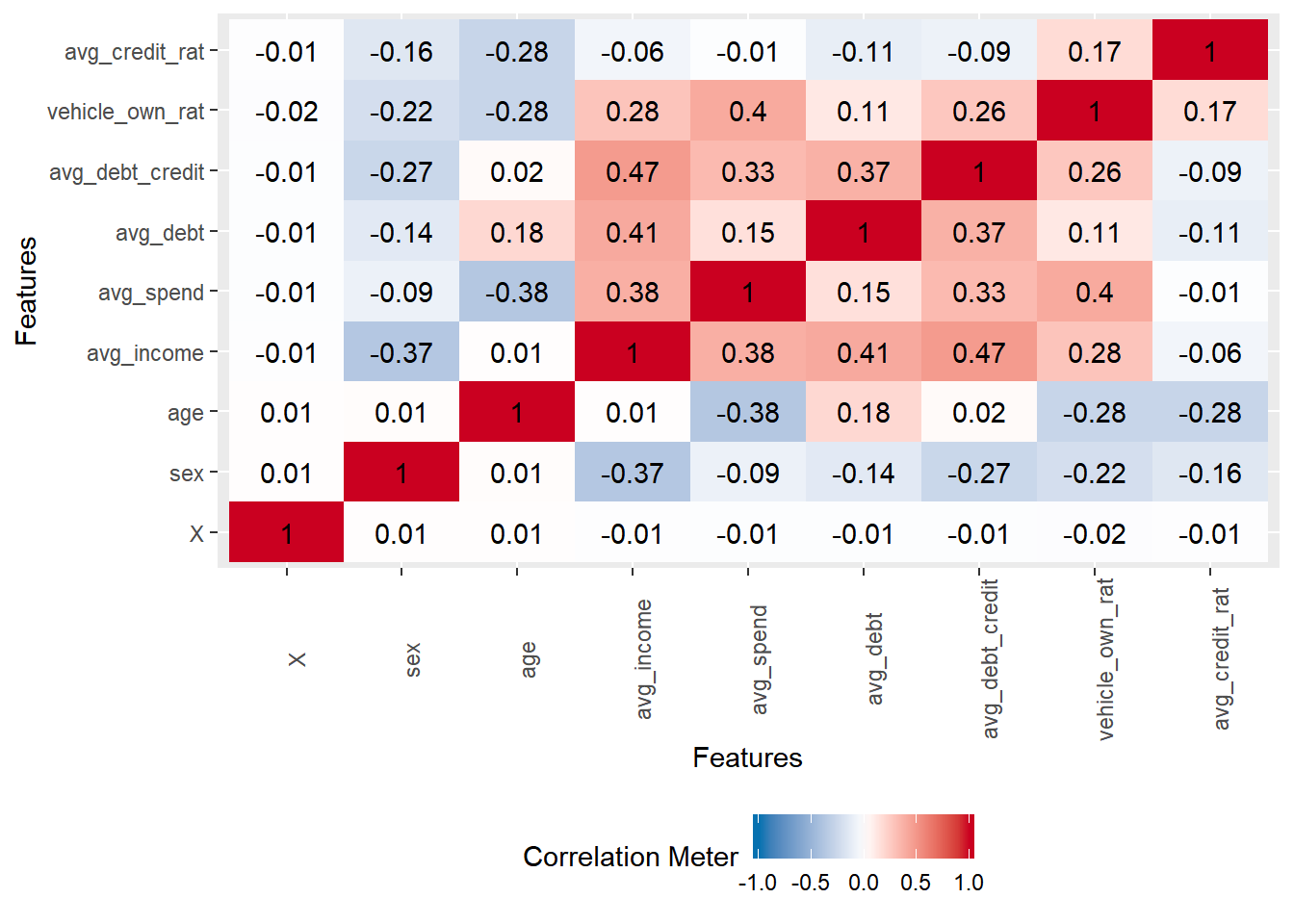
plot\_histogram(jeju\_data\_party)



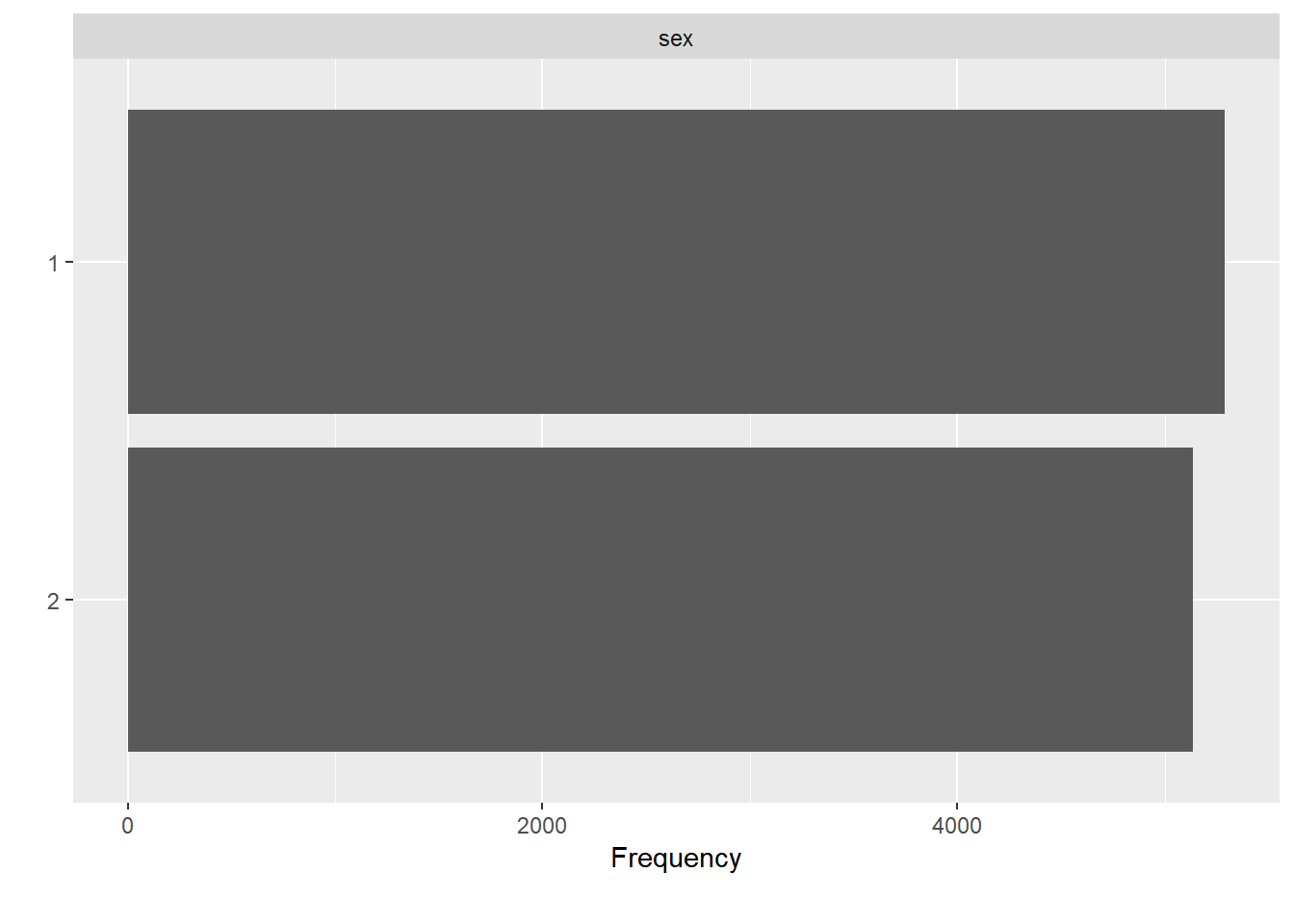
plot\_density(jeju\_data\_party)



plot\_correlation (jeju\_data\_party, type = 'continuous', 'jeju\_data\_party2')



plot\_bar(jeju\_data\_party)



#create\_report(jeju\_data\_party)  
  
arr\_delay\_df <- jeju\_data\_party  
plot\_boxplot(arr\_delay\_df, by = "avg\_credit\_rat")

