숭실대학교

과목명 : 빅데이터분산컴퓨팅

팀 명 : We Can Do IT

팀 원 : 정보통계보험수리학과 김민우

: 정보통계보험수리학과 이강현

: AI 융합학부 정지인

<목차>

1. 프로젝트 요약

본 프로젝트의 주제는 **<뉴욕택시의 바가지 요금 특성 파악>** 이다.

먼저 바가지 요금은 회귀분석과 클러스터링을 통해 결정한다. 시간, 거리, 요금 등을 통한 구체적인 기준으로 바가지 요금이라고 판단된 데이터가 실제로는 바가지 요금이 아니라 잘못 입력된 데이터일 수 있다. 이는 분석의 방향 및 목적에 가장 큰 전제오류가 될 수 있다. 따라서 이를 가능한 배제하기 위해 데이터의 정상적인 운행 기록의 기준을 보수적으로 잡았다. 나아가 데이터의 면밀한 탐색을 통해 이를 제거하도록 했다.

원래 데이터에서 Rate code가 2~4인 데이터는 JFK, Newark, Westchester 공항으로 운행요금을 정해놓고 운임하므로 분석에서 제외하며, Rate Code가 5인 데이터는 협상된 금액이므로 이 또한 분석 간 사용하지 않는다. 따라서 분석 방향에 맞는 Rate Code가 1인 Standard 경우에만 분석에 사용한다.

바가지 요금에 해당하는 데이터의 특성을 보다 명확하게 파악하기 위해 클러스터링 과정을 거쳤고, 클러스터 별로 바가지 요금의 특성을 분석하였다. 클러스터 별로 바가지 요금의 특성을 파악할 때는 중선형회귀를 이용하였다.

2. 프로젝트 선정배경 및 목적

> "美 뉴욕택시 바가지요금 만연"
>
> 연합
>
> 미국 뉴욕시의 택시기사들이 시내를 이동하는 승객들을 상대로 불법으로 시외 할증요금을 적용, 지난 2년간 830만달러(한화 약 93억6천500만원)가 넘는 요금을 더 받았다고 시 당국이 12일 밝혔다.
>
> 뉴욕시 택시.리무진위원회에 따르면 뉴욕에서 택시 영업 면허를 가진 운전사 4만8천300명 가운데 한 차례라도 이같은 부당요금을 징수한 운전사는 3만5천558명에 달한 것으로 조사됐다.

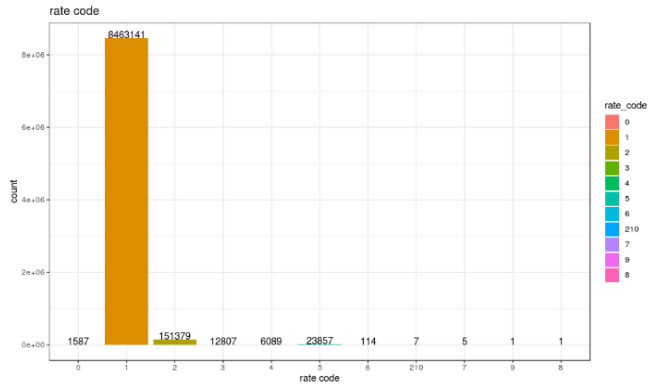
대다수의 사람들은 관광지에 여행을 가면 복잡한 대중교통보다는 편리한 택시를 선호한다. 그래서인지 현지 문화를 잘 모르는 관광객을 대상으로 하는 택시 바가지 요금에 대한 문제가 흔히 거론되었다. 바가지 요금은 개인에게 금전적인 피해를 줄 뿐만 아니라, 바가지 요금을 맞은 사람들은 관광지에 좋지 못한 기억을 갖게 된다. 이는 해당 관광지의 이미지를 실추시키며, 나아가 이를 방치하게 된다면 관광지의 수요 또한 하락하게 되는 결과를 초래한다.

2013년의 과거 데이터를 통해 바가지 요금이 씌워진 택시 운행 기록의 특성을 파악할 수 있다면 국가 및 기업 차원에서 특정 택시기사나 회사에게 조치를 취할 수 있고, 개인에겐 특정 장소나 시간대, 목적지에 대해서 표준 요금 정립 등을 통해 바가지 요금 방지에 대해 예방 차원의 도움을 줄 수 있을 것으로 예상한다.

3. 분석과정

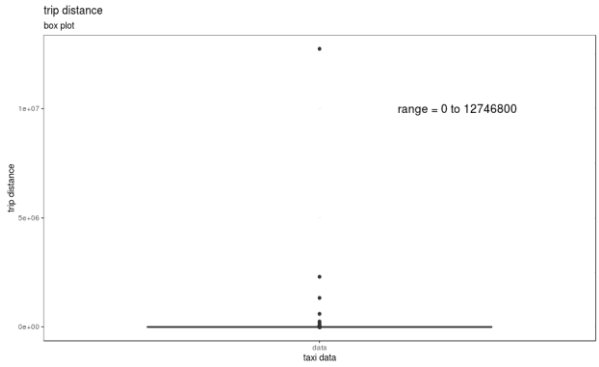
3.1. 데이터 전처리

▶ Rate Code



Rate Code 0, 7, 8, 9, 210인 데이터 제거

▶ Trip Distance

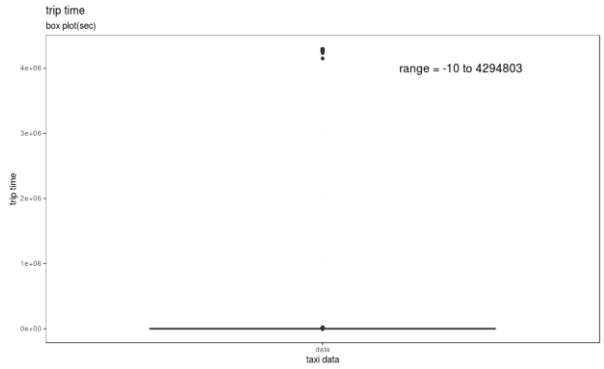


뉴욕 최장거리 횡단 시 60mile

0 mile 포함 이하의 값이나

100 mile 이상의 값 삭제

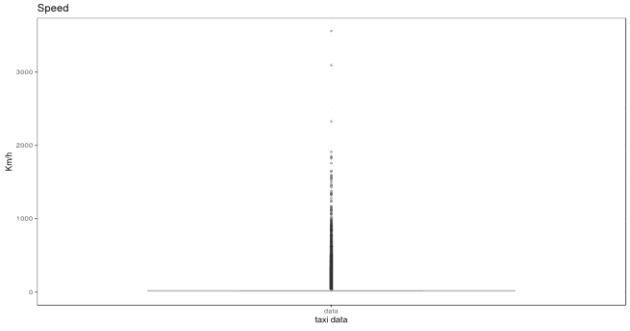
▶ Trip Time



Trip Time이 60 secs 이하이거나

4,000,000 secs 이상인 것은 삭제

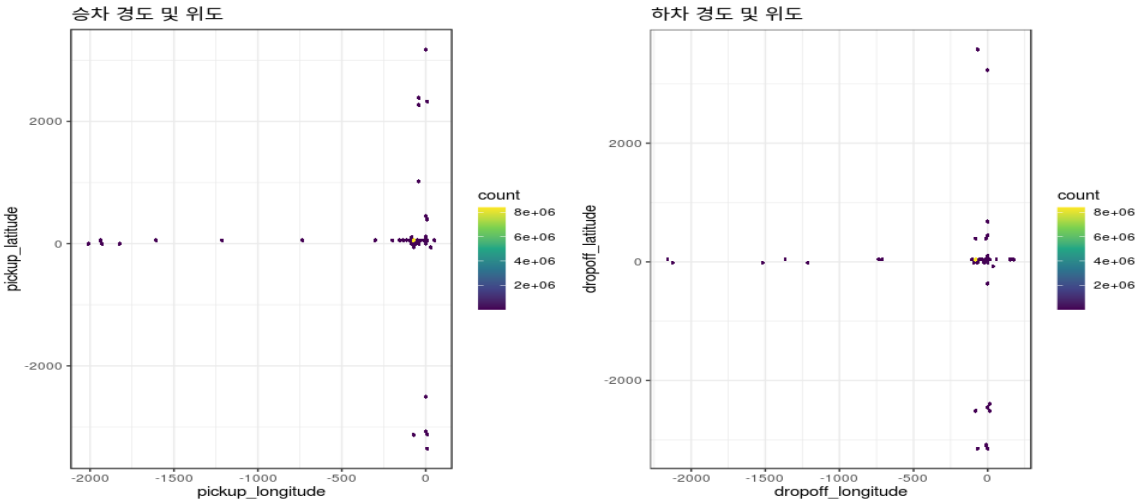
▶ Driving Speed



Time, Distance를 활용한 Speed 변수 생성

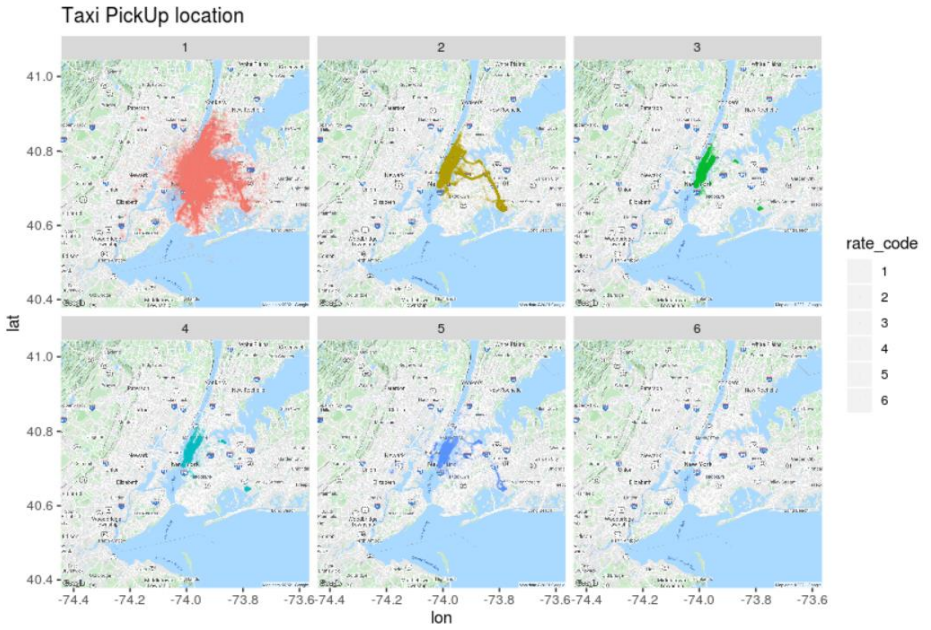
평균 속도 5 이하, 150 이상 제거 (Km/h)

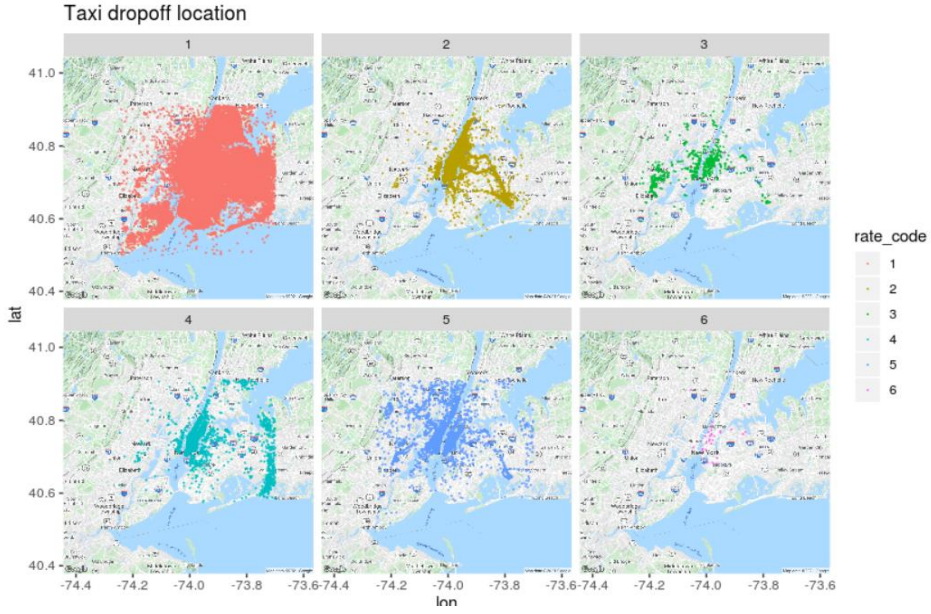
▶ Longitude, Latitude



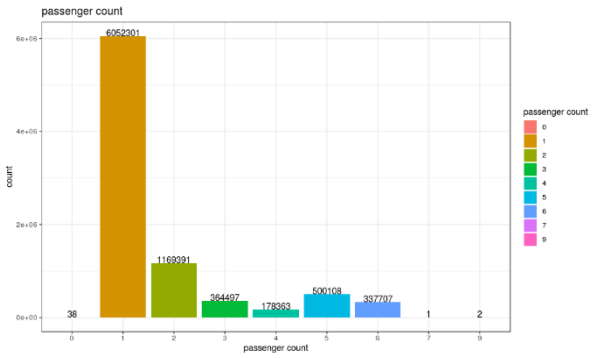
뉴욕 위도, 경도에 벗어나는 데이터 제거

➔ 제거 후 모습



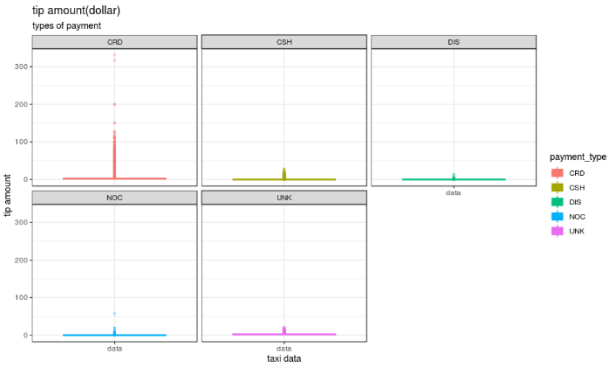


▶ Passenger Count



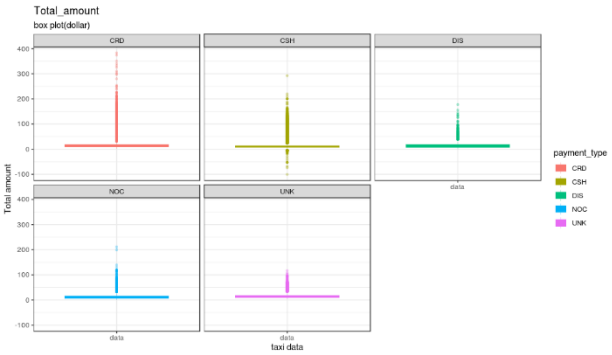
승객 수 0 명 데이터 제거

▶ Tip Amount



팁 금액 0 미만 데이터 제거

▶ Total Amount



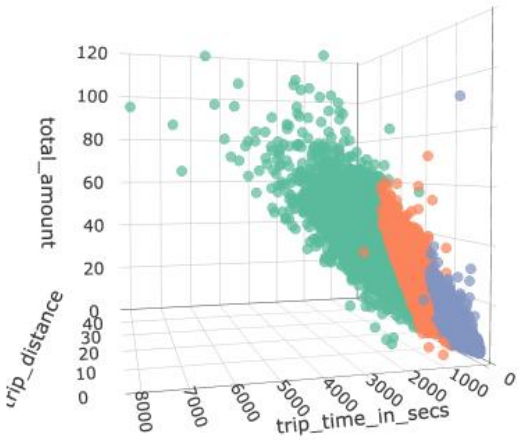
총 요금 0 이하 데이터 제거

3.2. 이상치 탐색

▶ K-means Clustering

약 8,000,000 개의 데이터 전체에서 바가지 요금에 대해 특성을 파악하는 것은 세부적인 특성을 보기엔 어려울 수 있다. 클러스터링 작업을 통해 군집으로 나눌 수 있다면 군집별로 보다 세부적인 특징 파악이 가능할 것으로 생각한다. 클러스터링에 사용한 변수는 여행 거리와 소요 시간, 금액을 사용했다.

3차원 K-means Clustering을 진행했다. 다음은 3차원 산점도와 군집 별 중앙 위치를 표로 나타냈다. 소요시간은 초 단위, 이동거리는 마일, 총 요금은 달러 단위 단위이다. K-means는 비지도학습 방법이기에 예측하기 쉽지 않지만 단거리, 중거리, 장거리 클러스터로 나누어진 것을 확인할 수 있다.



군집	소요시간	이동거리	총 요금
1	431.13	1.45	9.01
2	1221.74	4.37	20.88
3	2816.79	11.08	46.79

▶ Multiple Linear Regression

나누어진 3개의 클러스터별로 3번의 회귀분석을 진행했다. 종속변수는 총 요금, 독립변수는 소요시간과 이동시간을 사용했다. 각 군집 별 회귀계수와 design matrix를 이용해 각 군집 별 이동거리와 소요시간에 따른 정상금액의 추정이 가능하다. 정상금액이 추정 가능하려면 각 회귀분석의 적합도가 높아야 한단 전제와, 최소자승법은 이상치에 민감하므로 극단적인 이상치를 잘 정제했다는 전제가 필요하다. 변수 개수가 군집 별로 같으므로 수정된 결정계수를 사용하지 않고 결정계수를 통해 비교했으며, 극단적인 이상치는 전 단계에서 자체적인 기준으로 확실히 제거했다. 거리와 시간 변수는 다중공선성이 충분히 의심 가능하지만 분석 간 중요한 변수이므로 제거하지 않는다. p-value는 모두 < 2e-16 으로 유의하게 나왔으므로 따로 표에 첨부하지 않는다.

Common Formula : Total amount ~ Trip distance + Trip time in secs

군집	변수	회귀계수	결정계수 R^2
단거리	Intercept	3.142	
	Trip time in secs	0.00686	0.8329
	Trip distance	2.013	
중거리	Intercept	2.441	
	Trip time in secs	0.00641	0.8723
	Trip distance	2.449	
장거리	Intercept	4.783	
	Trip time in secs	0.00548	0.8379
	Trip distance	2.363	

4. 결과

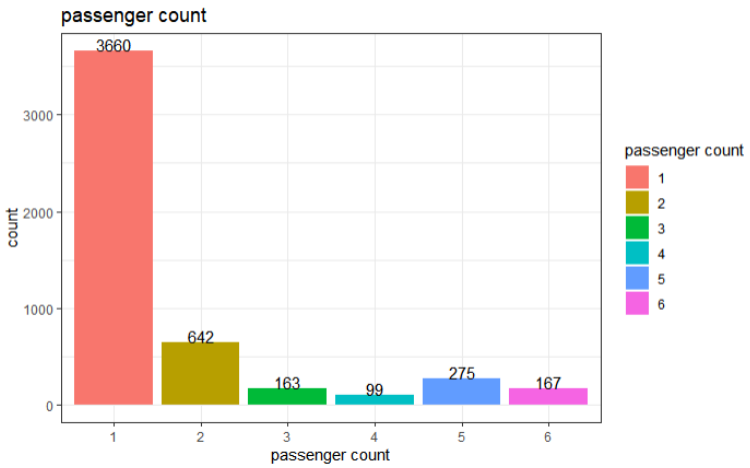
정상적인 데이터라고 보기 힘든 극단적인 이상치에 대한 전처리를 앞서 진행했다. 각 군집 별 회귀분석 결과를 통해 추정된 정상 금액에서 실제 비용을 뺀 잔차는, 정상적인 값보다 많이 지불한 금액이라고 생각할 수 있다. 잔차 중 양수인 것 중에 상위 0.01%에 대해 분석했다.

▶ **단거리, 특정 택시기사**에 대한 분석, 잔차 상위 0.01%, 5006개 데이터

hack_license <chr>	totalSum <dbl>	estimateSum <dbl>	ratio <dbl>	count <int>	countAll <int>
E971DEEC810C4A8D209F69E9749FC579	245.30	97.595584	2.513433	11	12
30207914030B55797A7930595208154 7	92.58	41.067822	2.254320	5	5
B62AB586B42EE9320A94E32B772C0FB3	163.42	105.868383	1.543615	5	128
2FA3B374D1F6FBD9A890A2E5BB0722A4	134.15	97.718821	1.372816	4	250
43347DCC012118624D7796BA983353E7	93.66	56.848995	1.647523	4	174
4A5C6A485FFC54B54B280DC61CADCF92	77.70	33.089421	2.348182	4	6
624F08173354FF95805B7B6C9318186D	91.65	55.166649	1.661330	4	182
E269AD1DAA16A17ED250D0BB4742D693	72.60	32.023689	2.267072	4	36
FAB0BEA22A1DBF52AB3F328A840AAB48	71.37	38.635244	1.847277	4	222
0057CCB5BA8D29E343B3D6D275AB22D3	56.42	21.964795	2.568656	3	205

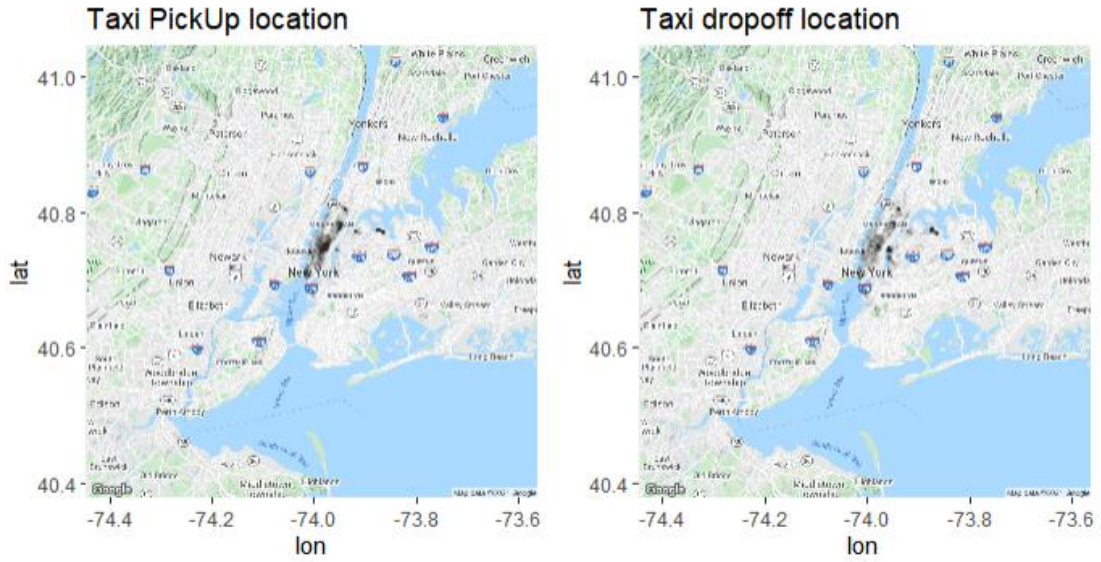
잔차 상위 0.01% 데이터 중 운행기록이 겹치는 택시기사가 11회, 5회, 4회 등으로 많이 존재한다. 5,000,000개의 데이터 중 잔차 상위 5,000개의 데이터에서 운행기록이 전체 12회 중 11회 겹친다는 것은 5,000개 데이터 무작위 추출에서의 확률과 비교해보면 매우 희박한 확률이며, 이는 분석의 타당성을 시사한다. 이 택시기사는 평균적으로 정상 추정 금액보다 2.5배 이상의 금액을 받았다. 정상 추정 요금 대비 실제 요금의 비율이 20배가 넘는 데이터 또한 다수 존재했다.

▶ **단거리, 승객 수**에 대한 분석, 잔차 상위 0.01%, 5006개 데이터



택시 바가지 요금은 승객이 1명일 때가 다른 경우보다 압도적으로 많았다.

▶ **단거리, 승 하차 위치**에 대한 분석, 잔차 상위 0.01%, 5006개 데이터



진하게 표시된 영역들은 주로 관광지/호텔에 속함.

- 라과디아 호텔
- Kenedy 다리
- 에스토리아 공원
- 엠파이어 스테이트
- 메트로폴리탄 미술관
- 센트럴파크
- 아폴로 영화관
- 유명 호텔 등

▶ **중거리, 특정 택시기사**에 대한 분석, 잔차 상위 0.01%, 2559개 데이터

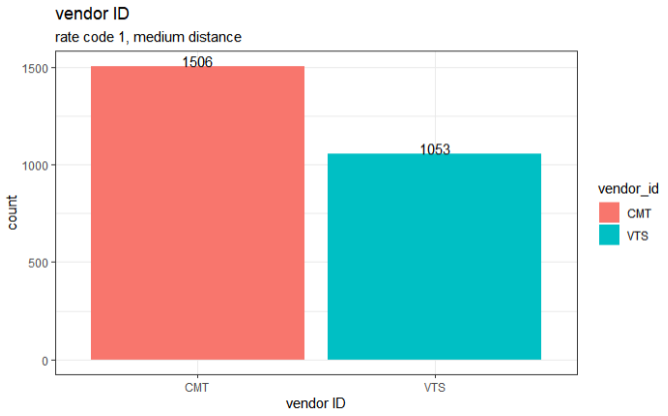
hack_license <chr>	totalSum <dbl>	estimateSum <dbl>	ratio <dbl>	count <int>	countAll <int>
E971DEEC810C4A8D209F69E9749FC579	423.92	134.740223	3.146202	12	12
7E2B7587F1A62A5021B57FF7BD8A0952	358.97	112.554684	3.189294	9	26
E07F31E8CEEFFB2F52966B6BE7512378	351.56	111.956980	3.140135	9	9
4A5C6A485FFC54B54B280DC61CADCF92	302.22	102.007018	2.962737	8	8
6D9460AA6A2083EE301E8ABC887245F3	308.39	99.001303	3.115010	8	8
A9B770584F117D21300312BB1B74D7CB	280.84	89.130632	3.150881	8	8
BF608E59A407739FCD5618312D35CB77	302.68	100.331682	3.016794	8	8
CB85E48145C28DA1DF194AC07EFD9076	298.91	103.939784	2.875800	8	8
1D105595DD510A7FC1AB006D11E4E1F0	260.20	86.925251	2.993376	7	7
B8C7594C95BB86A228C785305287583B	259.10	94.526028	2.741044	7	7

첫번째 기사는 단거리 특정 택시기사 분석에서와 같은 기사이며 단거리 상위 과금 데이터에서도 12회, 중거리 상위 과금 데이터에서도 12회를 운행했다. 단거리에서는 12회 중 11회, 중거리에선 12회 중 12회가 바가지 요금으로 나왔다. 바가지 요금을 빈번하게 씌우는 기사로 분석할 수 있다. 중거리에서는 평균 과금 비율이 3.14배로 단거리보다 증가했음을 확인할 수 있다. 4번째 택시기사 역시 단거리에서도 확인했던 택시기사이다. 단거리에선 6회 중 4회, 중거리에선 8회 중 8회 모두가 과금으로 확인됐다. 이 택시기사도 과금 비율이 증가했다.

▶ **중거리, 승객 수**에 대한 분석, 잔차 상위 0.01%, 2559개 데이터

1명인 경우가 2559회 중 1972회를 차지한다.

▶ **중거리, TPEP 제공 업체**에 대한 분석, 잔차 상위 0.01%, 2559개 데이터



CMT 사가 *VTS* 사보다 약 40% 많았다.

▶ **중거리, 승 하차 시간대**에한 분석, 잔차 상위 0.01%, 2559개 데이터

주로 새벽 시간대나 밤시간대에 많았다. 하지만 이는 새벽, 밤시간대의 할증 등의 잠재변수로 인한 과금일 수 있지만, 그럼에도 불구하고 많은 횟수를 차지하고 있다.

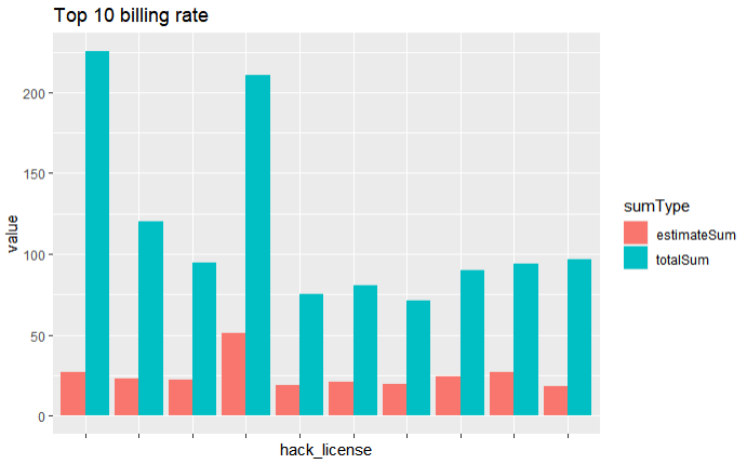
▶ **중거리, 승 하차 위치**에 대한 분석, 잔차 상위 0.01%, 2559개 데이터

단거리와 큰 차이를 보이지 않았으며 주로 관광지나 호텔에 위치했다.

▶ **장거리, 특정 택시 기사**에 대한 분석, 잔차 상위 0.01%, 564개 데이터

hack_license	totalSum	estimateSum	ratio	count	countAll
11174783A720D7A2EAE31B131C384F1D	274.12	121.59485	2.254372	5	34
E269AD1DAA16A17ED250D0B84742D693	290.64	87.50125	3.321553	5	19
E040FCD0820543D53E5BF71795364887	208.80	73.18913	2.852883	4	4
07E42629EDA15272D5304B00C72FECBF	209.33	67.00995	3.123864	3	3
0BE75C352825AE8C6F3279D65F74E45B	190.90	56.07485	3.404378	3	3
8CBB91C270C739C2EDCB801BE2D03B18	214.70	68.97315	3.112805	3	29
A7554972722C6F9A87141FCA591AE8C2	168.80	56.72868	2.975567	3	3
FB9E42ED1CD384EC3DCB67FDB36B8EC2	177.32	59.20003	2.995269	3	19
0D61304FB9E9E7CF6781EA04D1B1C30B	100.00	33.66146	2.970756	2	2
11F9C8E2B165160FCD8FCBC081392739	120.00	40.62462	2.953874	2	9

두번째 택시기사는 단거리에서도 확인 가능한 택시기사다. 단거리에선 36회 중 4회, 장거리에선 19회 중 5회를 과금했다. 단거리에서는 과금 비율이 약 2배였고 장거리에선 3.3배로 증가했음을 확인할 수 있다. 또 8배 이상의 금액을 과금한 기사들도 많았다.

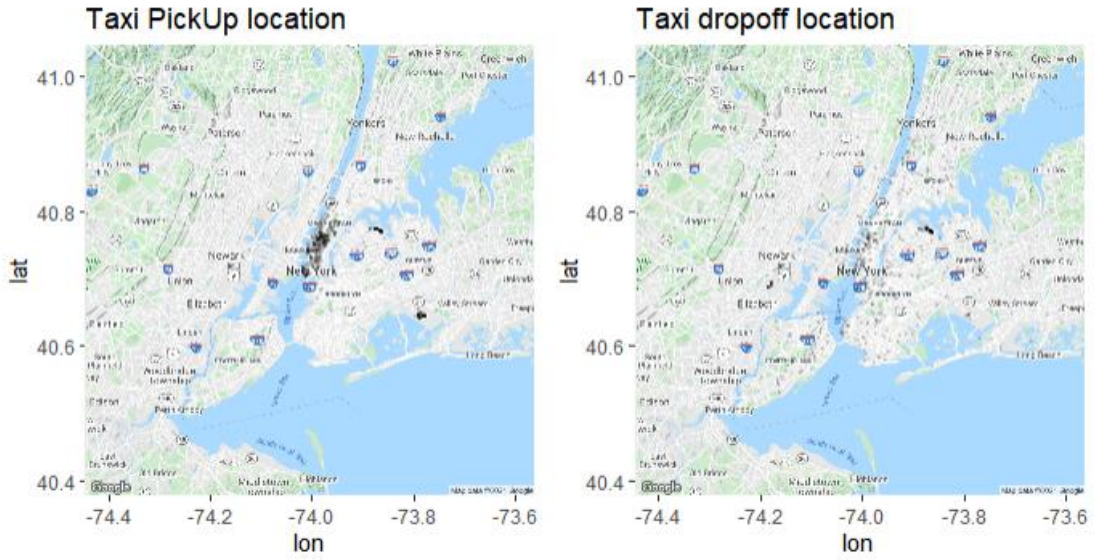


estimateSum(빨강색)은 과금 상위 10명의 택시기사 별 추정된 예상 정상 금액의 합산이며, totalSum(녹색)은 같은 기사의 실제 추징 요금의 합산이다. 추정된 정상 금액과 실제 요금의 차이가 매우 극명한 것을 확인할 수 있다.

▶ **장거리, 승객 수**에 대한 분석, 잔차 상위 0.01%, 564개 데이터

1명인 경우가 564개 중 405개를 차지한다.

▶ **장거리, 승 하차 위치**에 대한 분석, 잔차 상위 0.01%, 564개 데이터



엠파이어 스테이트 빌딩 근처에서 장거리 경우 과금이 많은 것은 여전하지만 탑승지가 JFK 공항인 경우가 추가됐다. 이는 뉴욕 여행객이 JFK 공항에 도착 후 멀리 이동했을 때인 것으로 예상된다.

5. 후기

▶ Lesson Learned

평소 개인 PC로는 다루기 어려운 대용량 데이터를 R studio server와 하둡 분산 파일 시스템(HDFS)을 이용해 직접 분석하면서 HDFS의 필요성을 느낄 수 있었다.

적절하게 하둡 분산 파일 시스템(HDFS)을 분석에 직접 사용하기 위해서는 HDFS의 동작 원리와 MapReduce의 사용 방법에 대한 깊은 이해가 필요했고, 이 부분을 팀원들이 함께 공부하며 개념을 단단하게 다질 수 있었다. 분석 방법 중 K-means와 회귀분석을 내장함수를 사용하지 않고 HDFS에서의 사용을 위해 직접 알고리즘을 구현하면서, 이 두 가지 분석 기술의 동작 원리 또한 공부가 필요했고, 이는 통계 공부에도 많은 도움이 되었다.

R 내장 함수의 편리함도 느낄 수 있었다. R-Hadoop에서도 여러가지 분석 알고리즘들을 R 내장 함수처럼 간단하게 사용할 수 있게 된다면 더욱 편리해질 것이라고 생각한다. 다방면에서 많은 배움을 얻을 수 있는 프로젝트였고, 특히 다른 과 팀원과의 협업은 다방면에서 도움을 주고 받을 수 있었으며 예상보다 긍정적인 효율을 가져왔다.

▶ Process

날짜	시간	장소	내용
11/29	14:00 ~ 17:00	용산역	자료 탐색, 아이디어 회의
12/01	09:00 ~ 09:45	zoom	교수님과의 면담, 아이디어 점검
12/01	23:00 ~ 01:00	zoom	프로젝트 방향 결정 및 분석 방법 수립
12/03	22:00 ~ 24:00	Zoom	데이터 전처리 후 결과 점검 및 수정
12/04	14:00 ~ 24:00	Zoom	군집화 작업 (k-means), 데이터 시각화
12/05	14:00 ~ 03:00	Zoom	Cluster별 회귀 모형 적합 및 바가지 데이터 판단
12/16, 17,18	15:00 ~ 03:00	Home, Zoom	바가지 데이터 분석 및 시각화, 보고서 작성

▶ Roles

이름	역할
김민우	● 데이터 전처리 및 시각화 ● Kmeans mapreduce 함수 목적에 맞게 수정 ● 데이터 분석 및 시각화 ● 보고서 작성 [결과]
이강현	● 자료조사 및 변수에 대한 이해 및 팀원 공유 ● Kmeans mapreduce 함수 목적에 맞게 수정 ● 데이터 분석 및 시각화 ● 보고서 작성 [본론]
정지인	● 이론조사 ● 바가지 요금에 대한 여러가지 가설 아이디어 제시 ● 회귀분석 mapreduce 함수 반환 값에 여러 통계치 추가 ● 보고서 작성 [서론]

6.부록 (R code)

- **사용 라이브러리**

```
library(dplyr)
library(ggplot2)
library(tidyverse)
library(lubridate)
library(patchwork)
library(plotly)
library(ggmap)
library(tidyr)
library(rhdfs)
library(rmr2)
```

- **회귀분석 MapReduce**

```
regmap <- function(., v) {
  dat <- data.frame(total_amount = v$total_amount, trip_distance = v$trip_distance, trip_time_in_secs = v$trip_time_in_secs)
  Xk <- model.matrix(total_amount ~ trip_distance + trip_time_in_secs, dat)
  yk <- as.matrix(dat[,1])
  XtXk <- crossprod(Xk, Xk)
  Xtyk <- crossprod(Xk, yk)
  ytyk <- crossprod(yk, yk)
  res <- list(XtXk, Xtyk, ytyk)
  keyval(1, res)
}
regreduce <- function(k, v) {
  XtX <- Reduce("+", v[seq_along(v) %% 3 == 1])
  Xty <- Reduce("+", v[seq_along(v) %% 3 == 2])
  yty <- Reduce("+", v[seq_along(v) %% 3 == 0])
  res <- list(XtX = XtX, Xty = Xty, yty = yty)
  keyval(1, res)
}

# summary : beta.hat과 MSE등 분석 결과 summary
fun.summary <- function(v) {
  XtX = v$XtX
  Xty = v$Xty
  yty = v$yty
  beta.hat = solve(XtX, Xty)
  nn = XtX[1,1]
  ysum = Xty[1]
  ybar = ysum/nn
  stat <- list(nn = nn, beta.hat = beta.hat, ysum = ysum, ybar = ybar)
  SSE = yty - crossprod(beta.hat, Xty)
  SST = yty - ysum^2/nn
  SSR = SST - SSE
  SS <- list(SSR = SSR, SSE = SSE, SST = SST)
  df.reg = dim(XtX)[1L] - 1
  df.tot = nn - 1
  df.res = df.tot - df.reg
  DF <- list(df.reg = df.reg, df.res = df.res, df.tot = df.tot)
  MSR = SSR / df.reg
  MST = SST / df.tot
  MSE = SSE / df.res
  MS <- list(MSR = MSR, MSE = MSE, MST = MST)
  f.val = MS$MSR / MS$MSE
  p.val = pf(f.val, DF$df.reg, DF$df.res, lower.tail = F)
  anova <- list(DF = DF, SS = SS, MS = MS, f.val = f.val, p.val = p.val)
  res <- list(mat = v, stat = stat, anova = anova)
}
```

- **K-means MapReduce**

```
dist.fun <- function(Center, Dat)
apply(Center, 1, function(x) colSums((t(Dat) - x)^2))
num.clusters = 3
C = NULL

kmeans.map <- function(., P) {
  nearest <- if(is.null(C)){
    sample(1:num.clusters, nrow(P), replace = TRUE)
  }
  else {
    D <- dist.fun(C, P)
    nearest <- max.col(-D)
  }
  keyval(nearest, cbind(nearest,P))
}
```

- **데이터 불러오기**

```
rmr.options(backend = "hadoop")
files <- hdfs.ls("/data/taxi/combined")$file

mr <- mapreduce(input = files[1],
                input.format = make.input.format(format = "csv", sep=",", stringsAsFactors=F)
)

res <- from.dfs(mr)
ress <- values(res)

colnames.tmp <- as.character(ress[,1])
class.tmp <- as.character(ress[,2])

colnames <- colnames.tmp[-1]
class <- class.tmp[-1]

class[c(6,8,9,10)] <- "numeric"

input.format <- make.input.format(
  format = "csv", sep = ",",
  stringsAsFactors = F,
  col.names = colnames, colClasses = class
  )

files <- files[-1]

data <- mapreduce(input = files,
input.format = input.format)
res.data <- from.dfs(data)
taxi <- res.data$val
dat <- taxi[,-12] # store_and_fwd_flag 변수 삭제
```

➢ **데이터 전처리**

```
## 1. rate code

n_of_obs <- dat %>% group_by(rate_code) %>%
summarize(count=n())

ggplot(n_of_obs,
       aes(x = as.factor(rate_code), y = count, fill
= rate_code)) +
  geom_bar(stat = "identity") +
  labs(x = "rate code",
       y = "count",
       title = "rate code") +
  geom_text(aes(label = count), vjust=0) +
  theme_bw()

## rate_code 변수가 1~6의 값이 아닌 것 삭제
dat <- dat[-which(dat$rate_code==0 |
dat$rate_code == 210 | dat$rate_code == 7 |
dat$rate_code == 9 | dat$rate_code == 8),]



## 2. trip_distance(mile)

dat %>% ggplot(aes(x = "data", y =
trip_distance)) +
  geom_boxplot() +
  labs(x = "taxi data",
       y = "trip distance",
       title = "trip distance",
       subtitle = "box plot") +
  theme_bw() +
  annotate("text", x = 1.3, y = 10000000, size = 5,
           label = paste0("range = ",
paste(as.character(range(dat$trip_distance)[1]),as.c
haracter(range(dat$trip_distance)[2]), sep= " to ")))

## 뉴욕주 최장거리로 횡단 시 60 mile,  0 mile
이하의 값이나 100 mile 이상의 값 삭제

dat <- dat[-which(dat$trip_distance <= 0 |
dat$trip_distance >= 100),]
summary(dat$trip_distance)

## 3. trip_time_in_secs

dat %>% ggplot(aes(x = "data", y =
trip_time_in_secs)) +
  geom_boxplot() +
  labs(x = "taxi data",
       y = "trip time",
       title = "trip time",
       subtitle = "box plot(sec)") +
  theme_bw() +
  annotate("text", x = 1.3, y = 4000000, size = 5,
           label = paste0("range = ",
paste(as.character(range(dat$trip_time_in_secs)[1]),
as.character(range(dat$trip_time_in_secs)[2]), sep=
" to ")))

## trip_time_in_secs이 60 이하이거나 4,000,000
이상인 것들 삭제
dat <- dat[-which(dat$trip_time_in_secs <= 60 |
dat$trip_time_in_secs >= 4000000), ]



## 4. passenger_count
n_of_obs <- dat %>%
group_by(passenger_count) %>%
summarize(count=n())



ggplot(n_of_obs,
       aes(x = as.factor(passenger_count), y =
count, fill = as.factor(passenger_count))) +
  geom_bar(stat = "identity") +
  labs(x = "passenger count",
       y = "count",
       title = "passenger count",
       fill = "passenger count") +
  geom_text(aes(label = count), vjust=0) +
  theme_bw()

## 승객 수가 0명 이하인 것 제외
dat <- dat[-which(dat$passenger_count<= 0) ,]



## 5. longitude, latitude

g1 <- dat %>% ggplot(aes(x = pickup_longitude,
y = pickup_latitude)) +
  geom_hex(bins=100) +
  scale_fill_continuous(type = "viridis") +
  labs(title = "승차 경도 및 위도") +
  theme_bw()

g2 <- dat %>% ggplot(aes(x = dropoff_longitude,
y = dropoff_latitude)) +
  geom_hex(bins=100) +
  scale_fill_continuous(type = "viridis") +
  labs(title = "하차 경도 및 위도") +
  theme_bw()

g1 + g2



## 뉴욕주 경도 범위 -74.2 ~ -73.5
## 뉴욕주 위도 범위 40.45 ~ 40.95
## 뉴욕주 위도 경도에서 벗어나는 점들 제외

dat <- dat[-which(dat$pickup_longitude <= -74.2
| dat$pickup_longitude >= -73.5) ,]
dat <- dat[-which(dat$pickup_latitude <= 40.45 |
dat$pickup_latitude >= 40.95) ,]
dat <- dat[-which(dat$dropoff_longitude <= -
74.2 | dat$dropoff_longitude >= -73.5) ,]
dat <- dat[-which(dat$dropoff_latitude <= 40.45 |
dat$dropoff_latitude >= 40.95) ,]

ggmap::register_google(key = "API Key")
ggmap::has_google_key()
NY <- ggmap::get_map("New York, NY, USA",
maptype = "terrain")

g3 <- ggmap(NY) + geom_point(data = dat,
                             aes(x =
pickup_longitude, y = pickup_latitude, color =
rate_code), size =.03, alpha =.1) +
  labs(title = "Taxi PickUp location") +
  facet_wrap(~rate_code)
g3

g4 <- ggmap(NY) + geom_point(data = dat,
                             aes(x =
dropoff_longitude, y = dropoff_latitude, color=
rate_code), size =.05) +
  labs(title = "Taxi dropoff location") +
  facet_wrap(~rate_code)
g4

g3 + g4



## 6. tip_amount

dat %>% ggplot(aes(x = "data", y = tip_amount,
fill=payment_type, color= payment_type)) +
  geom_boxplot(outlier.size = 1, outlier.alpha
= .3, notchwidth = 0.01) +
  labs(x = "taxi data",
       y = "tip amount",
       title = "tip amount(dollar)",
       subtitle = "types of payment") +
  theme_bw() +
  facet_wrap(~payment_type)

## tip이 음수인 것 제외
dat <- dat[-which(dat$tip_amount < 0), ]



## 7. total_amount

dat %>% ggplot(aes(x = "data", y =
total_amount, fill=payment_type, color=
payment_type)) +
  geom_boxplot(outlier.size = 1, outlier.alpha
= .3, lwd=0.1, notch = T) +
  labs(x = "taxi data",
       y = "Total amount",
       title = "Total_amount",
       subtitle = "box plot(dollar)") +
  theme_bw() +
```

```
  facet_wrap(~payment_type)

## 총 금액이 0 이하인 것 제외
dat <- dat[-which(dat$total_amount <= 0), ]


## 8. 속력 변수 km/h

dat.res <- dat %>% select(trip_distance,
trip_time_in_secs) %>% mutate(km_per_hour =
5793.64*trip_distance/trip_time_in_secs)

dat.res %>% ggplot(aes(x = "data", y =
km_per_hour)) +
  geom_boxplot(outlier.size = 1, outlier.alpha
= .3, lwd=0.1, notch = T) +
  labs(x = "taxi data",
       y = "Km/h",
       title = "Speed") +
  theme_bw()

## 5 이하 150 이상 제거
dat <- dat[-which(dat.res$km_per_hour <= 5 |
dat.res$km_per_hour >= 150), ]
```

➢ **K-means 함수 사용**

```
max.iter <- 1:10

for(i in seq_along(max.iter)){
  mr <-
from.dfs(mapreduce( to.dfs(dat[,c("trip_time
_in_secs", "trip_distance", "total_amount")]),
map = kmeans.map))
  C <- aggregate(x = mr$val, by =
list(mr$key), FUN = mean)[,-c(1,2)]; C
}
```

➢ **데이터 클러스터 번호 열 추가**

```
s.mr <- arrange(mr$val, trip_time_in_secs,
total_amount, trip_distance)
s.dat <- arrange(dat,
trip_time_in_secs,total_amount,trip_distance)

clusterNum <- s.mr %>% select(nearest)

taxiWithGroup <- s.dat %>% mutate(cluster
= clusterNum[,1])
```

➢ **데이터 클러스터 별로 나누기**

```
taxiG1 <-
taxiWithGroup[which(taxiWithGroup$cluster
== 3), ] %>% select(-cluster) # 단거리
taxiG2 <-
taxiWithGroup[which(taxiWithGroup$cluster
== 1), ] %>% select(-cluster) # 중거리
taxiG3 <-
taxiWithGroup[which(taxiWithGroup$cluster
== 2), ] %>% select(-cluster) # 장거리
```

➢ **단거리 분석**

```
# 중선형회귀 P=2
reg.result <-
values(from.dfs(mapreduce(to.dfs(taxiG1),
map = regmap, reduce = regreduce,
combine = T)))
reg.summary <- fun.summary(reg.result)

# 잔차가 큰 값 선택

# 디자인 행렬
taxiDesingMat <- model.matrix(object =
total_amount ~ trip_distance +
trip_time_in_secs, data =
taxiG1[,c("trip_distance", "trip_time_in_secs",
"total_amount")])

# 추정치 계산
fittedFare <-
t(reg.summary$stat$beta.hat) %*%
t(taxiDesingMat)
fittedFare <- t(fittedFare)


resi <- taxiG1$total_amount - fittedFare[,1]
# 잔차
BigRes <- sort(resi, decreasing = T) %>%
names() # 잔차 INDEX
BigResIDX <-
BigRes[1:ceiling(length(BigRes)*0.001)] # 잔
차 INDEX 중 상위 0.1%


# 실제 낸 돈 - 추정한 정상금액 == 잔차
taxiG1[BigResIDX,] %>%
mutate(estimateFare =
fittedFare[,1][BigResIDX], residual =
resi[BigResIDX]) %>% select(total_amount,
estimateFare, residual) %>% head()
taxiG1[BigResIDX,] %>%
mutate(estimateFare =
fittedFare[,1][BigResIDX], residual =
resi[BigResIDX]) %>% select(total_amount,
estimateFare, residual) %>% tail()


# 변수가 다 있는 데이터로 돌아옴
selectedDataG1 <- taxiG1[BigResIDX, ]

# 가설 1
# VendorID : 특정 업체에서 유독 많다
selectedDataG1 %>%
select(vendor_id) %>% table()
nOfObs <- selectedDataG1 %>%
group_by(vendor_id) %>%
summarize(count = n())

ggplot(nOfObs,
       aes(x = as.factor(vendor_id), y =
count, fill = vendor_id)) +
  geom_bar(stat = "identity") +
  labs(x = "vendor ID",
       y = "count",
       title = "vendor ID",
       subtitle = "rate code 1, short
distance") +
  geom_text(aes(label = count), vjust=0) +
  theme_bw()

# CMT사가 VTS 사에 비해 많다.


# 가설 2
# 덤티기를 씌우는 기사들은 특징이 있을
것이다.
BadDrivers <- selectedDataG1 %>%
select(hack_license, total_amount) %>%
mutate(estimateFare =
fittedFare[,1][BigResIDX]) %>%
group_by(hack_license) %>%
summarize(totalSum = sum(total_amount),
estimateSum = sum(estimateFare), count =
n()) %>% mutate(ratio =
totalSum/estimateSum) %>%
arrange(desc(count))
BadDrivers %>% arrange(desc(ratio))


allDrivingFreqG1 <- taxiG1 %>%
select(hack_license) %>%
group_by(hack_license) %>%
```

```
summarize(count = n())
bigResDrivingFreaG1 <-
selectedDataG1 %>%
select(hack_license) %>%
group_by(hack_license) %>%
summarize(count=n())

count <- inner_join(allDrivingFreqG1,
bigResDrivingFreaG1, by = "hack_license",
suffix = c("All", "Outlier")) %>%
arrange(desc(countOutlier))
BadDriversDone <- bind_cols(BadDrivers,
count[,"countAll"])[,
c("hack_license","totalSum", "estimateSum",
"ratio", "count", "countAll")]

BadDriversDone %>%
arrange(desc(count)) %>% head(20)


pivotBadDrivers <- BadDriversDone %>%
select(hack_license, totalSum, estimateSum,
ratio) %>% arrange(desc(ratio)) %>%
head(10) %>% select(-ratio) %>%
pivot_longer(cols = !hack_license, names_to
= "sumType", values_to = "value")

ggplot(pivotBadDrivers, aes(x =
hack_license, y = value, fill = sumType)) +
  geom_col(position = "dodge") +
theme(axis.text.x = element_blank()) +
labs(title = "Top 10 billing rate")



# datetime : 특정시간대인지
pickupHourG1 <-
hour(selectedDataG1$pickup_datetime) %>
% table() %>% as.data.frame()
names(pickupHourG1) <- c("Hour", "count")

dropoffHourG1 <-
hour(selectedDataG1$dropoff_datetime) %>
% table() %>% as.data.frame()
names(dropoffHourG1) <- c("Hour",
"count")

hourG1 <- inner_join(pickupHourG1,
dropoffHourG1, by="Hour", suffix =
c("Pickup","Dropoff"))
hourG1 <- hourG1 %>% pivot_longer(cols
= !Hour, names_to = "hourType", values_to
= "value")
# Change line colors by groups
ggplot(hourG1, aes(x=Hour, y=value,
group=hourType)) +
  geom_line(aes(color=hourType), size =
1)+
  geom_point(aes(color=hourType), size =
2)



# Passenger count : 특정 승객수인지
NofObs <- selectedDataG1 %>%
group_by(passenger_count) %>%
summarize(count=n())


ggplot(NofObs,
       aes(x = as.factor(passenger_count),
y = count, fill =
as.factor(passenger_count))) +
  geom_bar(stat = "identity") +
  labs(x = "passenger count",
       y = "count",
       title = "passenger count",
       fill = "passenger count") +
  geom_text(aes(label = count), vjust=0) +
  theme_bw()
# 1명일때가 많다.


# 특정 승하차 위치
ggmap::register_google(key = "API KEY")
ggmap::has_google_key()

NYMAP <- ggmap::get_map("New York, NY,
USA", maptype = "terrain")

g3 <- ggmap(NYMAP) + geom_point(data
= selectedDataG1, aes(x =pickup_longitude,
y = pickup_latitude), size =.03, alpha =.05)
+ labs(title = "Taxi PickUp location")

g4 <- ggmap(NYMAP) + geom_point(data
= selectedDataG1, aes(x =
dropoff_longitude, y = dropoff_latitude),
size =.03, alpha=.05) +
  labs(title = "Taxi dropoff location")
g3 + g4
```

```
# 밀집된 위경도

# (1)
# 40.77411263879798, -73.87177711031461
# East Elmhurst (지역)
# 뉴욕 라과디아 공항 + 몇 유명 호텔


# (2)
# 40.778186, -73.925555
# 관광명소
# kenedy bridge + 에스토리아 공원


# (3)
# 40.7489662320195, -73.98564147146332
# 관광명소
# 엠파이어 스테이트 빌딩

# (4)
#   40.77995075048802, -
73.96316396457098
# 관광명소
# 메트로폴리탄 미술관, 솔로몬 R, 구겐하
임 미술관, 센트럴파크


# (5)
# 40.81007194617472, -73.95006688207894
# 관광명소
# 아폴로 영화관, 알로프트 할렘, 마르쿠스
가비공원


# Payment type : 특정 지불 방식인지
NofObs <- selectedDataG1 %>%
group_by(payment_type) %>%
summarize(count=n())


ggplot(NofObs,
       aes(x = as.factor(payment_type), y =
count, fill = as.factor(payment_type))) +
  geom_bar(stat = "identity") +
  labs(x = "payment type",
       y = "count",
       title = "payment type",
       fill = "payment type") +
  geom_text(aes(label = count), vjust=0) +
  theme_bw()
```

대부분이 카드 데이터에 있다.
아마 현금으로 결제한 덤티기에 대해서는 입력을 기사가 일부러 안할 수도 있다

TIp amount : 팁이 비싼지?
selectedDataG1[-which(selectedDataG1$tip_amount == 0),] %>% mutate(tipratio = tip_amount / total_amount) %>% select(tipratio) %>% summary()

cluster 1 (전체데이터) 에서 팁의 비율 (= 일반적인 팁의 비율, 약 15%)
taxiG1[-which(taxiG1$tip_amount == 0),] %>% mutate(tipratio = tip_amount / total_amount) %>% select(tipratio) %>% summary()

전체 데이터에서의 tip Ratio가 15%인데, 이상치로 판별된 데이터 셋의 tip Ratio는 15%보다 한참 큰 34.3% 이다(평균, tip=0 제외).
total amount에는 tip이 포함되지 않았음에도 불구하고.
미터기 금액도 많이 내고, 팁도 많이 내고

- **중거리, 장거리 분석**

위의 단거리 분석에서 변수명만 수정해서 사용 가능 000G1 -> 000G2 or G3