R_HW1_엄상준

우리는 gavote 라는 데이터를 가지고 Linear Regression Modeling 을 하여 분석할 것이다.

1. Data 불러오기

우리는 "faraway" package 안에 있는 gavote 데이터를 이용할 것이다.

data 명령어는 R 에 특정 dataset 을 불러와주는 역할을 한다.

그리고 library 를 통해 faraway package 를 불러올 수 있으며 데이터나 명령어 등에 대해서 궁금한 점들이 있다면, help, help.search 등의 명령어를 통해서 데이터나 명령어 등에 대한 설명을 찾을 수 있다.

```
data(gavote, package='faraway')
library(faraway)
help(gavote)

help(quantile)
help.search('quantiles')
```

2. 데이터 살펴보기

2-1. Overview

이제 데이터를 한 번 살펴보자.

gavote

```
> gavote
             equip   econ perAA rural    atlanta   gore   bush other  votes ballots
APPLING      LEVER   poor 0.182 rural notAtlanta   2093   3940    66   6099    6617
ATKINSON     LEVER   poor 0.230 rural notAtlanta    821   1228    22   2071    2149
BACON        LEVER   poor 0.131 rural notAtlanta    956   2010    29   2995    3347
BAKER        OS-CC   poor 0.476 rural notAtlanta    893    615    11   1519    1607
BALDWIN      LEVER middle 0.359 rural notAtlanta   5893   6041   192  12126   12785
BANKS        LEVER middle 0.024 rural notAtlanta   1220   3202   111   4533    4773
BARROW       OS-CC middle 0.079 urban notAtlanta   3657   7925   520  12102   12522
BARTOW       OS-PC middle 0.079 urban    Atlanta   7508  14720   552  22780   23735
```

Help 를 통해 살펴본 데이터의 설명으로는 이 데이터는 미국 조지아 주의 대통령 선거에 관한 데이터이다. 각각의 데이터는 미국 조지아의 County 를 의미하며 총 10 개의 variable 로 되어 있다.

equip: 투표 시 사용된 장비의 종류

econ: county 의 경제적 지위

perAA: county 내의 African American 의 비율

rural: 시골인지 도시인지 여부

atlanta: county 가 atlanta 에 속해 있는지 아닌지

gore: Gore 에 대한 득표수

bush: Bush 에 대한 득표수

other: 다른 candidate 에 대한 득표수

votes: 총 투표 수

ballots: 총 ballots 수

여기서 ballot 과 vote 의 차이점은 ballot 은 투표권이고 votes 는 투표권을 가지고 투표를 한 경우에 발생한다. 즉, 투표권이 있더라도 투표를 하지 않는다면, 또는 실패한다면 vote 는 일어나지 않는다.

```
head(gavote)
```

```
##          equip   econ perAA rural    atlanta gore bush other votes ballots
## APPLING  LEVER   poor 0.182 rural notAtlanta 2093 3940    66  6099    6617
## ATKINSON LEVER   poor 0.230 rural notAtlanta  821 1228    22  2071    2149
## BACON    LEVER   poor 0.131 rural notAtlanta  956 2010    29  2995    3347
## BAKER    OS-CC   poor 0.476 rural notAtlanta  893  615    11  1519    1607
## BALDWIN  LEVER middle 0.359 rural notAtlanta 5893 6041   192 12126   12785
## BANKS    LEVER middle 0.024 rural notAtlanta 1220 3202   111  4533    4773
```

Head 명령어를 이용하면 데이터를 전부 보지 않고 앞의 몇 개의 데이터만 확인할 수 있다.

```
str(gavote)
```

```
## 'data.frame':    159 obs. of  10 variables:
##  $ equip  : Factor w/ 5 levels "LEVER","OS-CC",..: 1 1 1 2 1 1 2 3 3 2 ...
##  $ econ   : Factor w/ 3 levels "middle","poor",..: 2 2 2 2 1 1 1 1 2 2 ...
##  $ perAA  : num  0.182 0.23 0.131 0.476 0.359 0.024 0.079 0.079 0.282 0.10
7 ...
##  $ rural  : Factor w/ 2 levels "rural","urban": 1 1 1 1 1 1 2 2 1 1 ...
##  $ atlanta: Factor w/ 2 levels "Atlanta","notAtlanta": 2 2 2 2 2 2 2 1 2
2 ...
##  $ gore   : int  2093 821 956 893 5893 1220 3657 7508 2234 1640 ...
##  $ bush   : int  3940 1228 2010 615 6041 3202 7925 14720 2381 2718 ...
##  $ other  : int  66 22 29 11 192 111 520 552 46 52 ...
```

```
##  $ votes  : int  6099 2071 2995 1519 12126 4533 12102 22780 4661 4410 ...
##  $ ballots: int  6617 2149 3347 1607 12785 4773 12522 23735 5741 4475 ...
```

앞선 명령어들이 데이터를 있는 그대로 보여주었다면, str 명령어는 데이터를 대략적으로 분석해서 보여준다. 예를 들어 총 관측치의 개수, 각 변수들의 type, level 수 등을 알려준다.

이 중 factor 데이터는 equip, econ, rural, atlanta 총 네 개라는 것을 알 수 있다.

```
summary(gavote)
```

```
##    equip        econ        perAA          rural          atlanta
##  LEVER:74   middle:69   Min.   :0.0000   rural:117   Atlanta   : 15
##  OS-CC:44   poor  :72   1st Qu.:0.1115   urban: 42   notAtlanta:144
##  OS-PC:22   rich  :18   Median :0.2330
##  PAPER: 2               Mean   :0.2430
##  PUNCH:17               3rd Qu.:0.3480
##                         Max.   :0.7650
##       gore             bush            other            votes
##  Min.   :   249   Min.   :   271   Min.   :   5.0   Min.   :   832
##  1st Qu.:  1386   1st Qu.:  1804   1st Qu.:  30.0   1st Qu.:  3506
##  Median :  2326   Median :  3597   Median :  86.0   Median :  6299
##  Mean   :  7020   Mean   :  8929   Mean   : 381.7   Mean   : 16331
##  3rd Qu.:  4430   3rd Qu.:  7468   3rd Qu.: 210.0   3rd Qu.: 11846
##  Max.   :154509   Max.   :140494   Max.   :7920.0   Max.   :263211
##     ballots
##  Min.   :   881
##  1st Qu.:  3694
##  Median :  6712
##  Mean   : 16927
##  3rd Qu.: 12251
##  Max.   :280975
```

Summary 는 데이터의 numerical overview 를 보여준다. 예를 들어 factor variable 의 경우 각 level 에 해당되는 데이터의 수는 몇 개인지, 또는 numerical variable 의 경우 4 분위 값과 최소 최대 값 등을 알려준다.

여기서 알 수 있는 점은 각 county 의 인구수가 제각각이기 때문에 votes 수와 ballots 등의 숫자가 그에 따라 차이가 많이 난다는 점이다.

2-2. Undercount data

우리가 이번 데이터에서 분석하고자 하는 것은 각 county 의 예상 Undercount 이다. Undercount 라는 것은 투표권에서 실제 투표수를 제한 것이다. 그런데 앞서 우리는 county 인구 수에 따라서 undercount 의 크기가 제각각일 것임을 알 수 있었다. 따라서 좀 더 정확한 분석을 위해 우리는 undercount 의 절대적 크기가 아닌 상대적 크기를 고려하는 것이 분석에 더 정확할 것이다.

이에 따라 변수를 생성하면,

```
gavote$undercount <- (gavote$ballots-gavote$votes)/gavote$ballots
```

또한 Undercount 의 4 분위 수는 다음과 같다.

```
summary(gavote$undercount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.02779 0.03983 0.04379 0.05647 0.18812
```

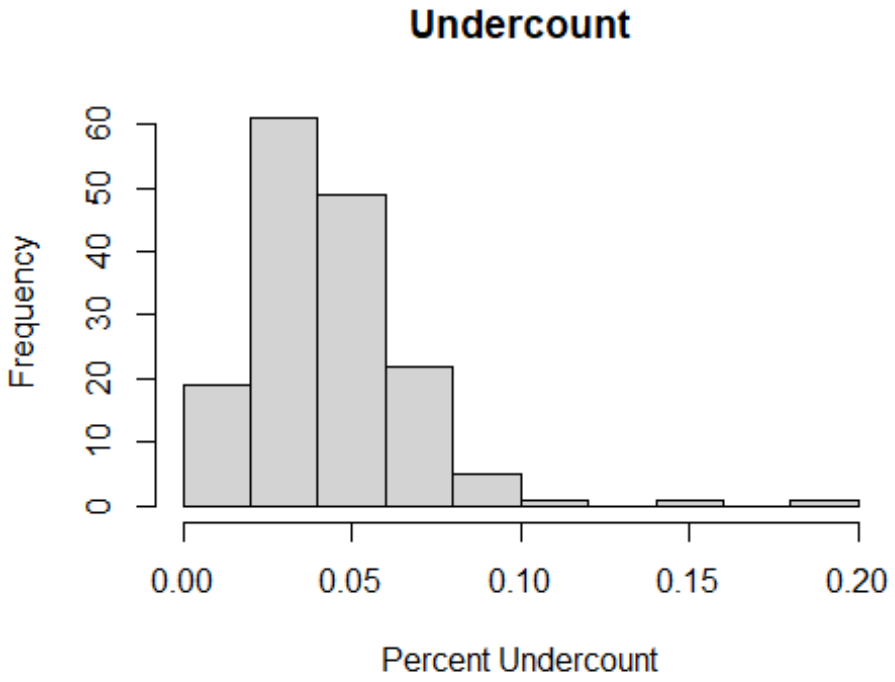
이 때 평균이 0.043 인데, 중요한 것은 overall relative undercount 와는 차이가 있다는 점이다.

```
with(gavote, sum(ballots-votes)/sum(ballots))
```

```
## [1] 0.03518021
```

Graphical Summary 는 변수에 대한 이해를 돕는 데에 사용된다.

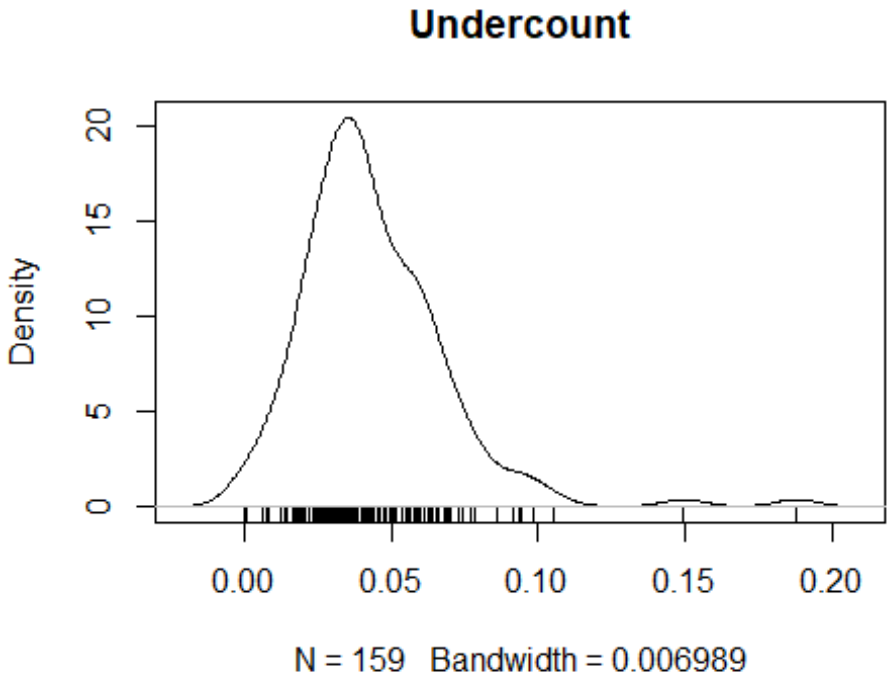
```
hist(gavote$undercount, main='Undercount', xlab='Percent Undercount')
```



Histogram 을 통해 살펴본 결과 오른쪽의 outlier 들로 인해 조금 skewed 된 모양이라는 것을 확인할 수 있다.

만약 bins 의 선택으로 인해 그래프 모양이 민감하게 변하는 것이 싫다면 Kernel Density Estimate 방법을 사용할 수도 있다.

```
plot(density(gavote$undercount), main="Undercount")
rug(gavote$undercount)
```



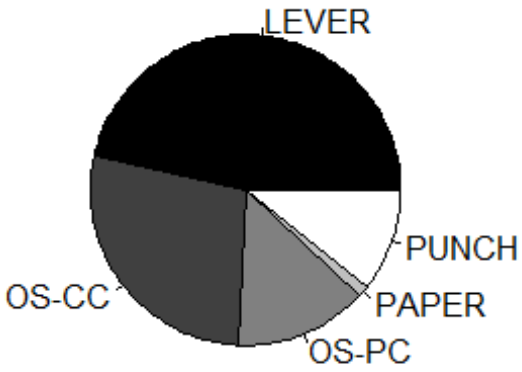
➔ 그래프의 모양이 좀 더 smooth 해진 것을 확인할 수 있다.

2-3. categorical data visualization

이번에는 categorical variable 들을 시각화하여 살펴보자.

Pie 명령어를 통해 각 level 들의 비중이 어느 정도 되는 지를 시각적으로 살펴볼 수 있다.

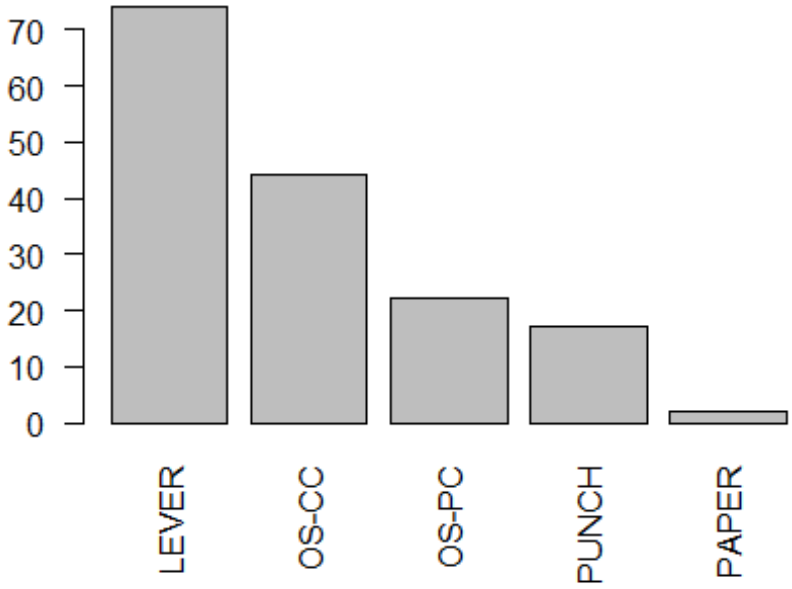
```
pie(table(gavote$equip), col=gray(0:4/4))
```



하지만 pie chart 는 상대적인 비중에 대한 비교가 조금 어려울 수 있다.

그래서 우리는 대신 Pareto Chart 를 이용할 수 있으며 barplot 명령어를 통해 이를 나타낼 수 있다.

```
barplot(sort(table(gavote$equip), decreasing = TRUE), las=2) ##las=2 bar label 들이 세로로 나타나게 함.
```



2-4. Scatter Plot

두 Numerical 또는 Quantitative 변수에 관해서는 scatter plot 을 그려서 두 변수 간의 관계를 알아볼 수 있다.

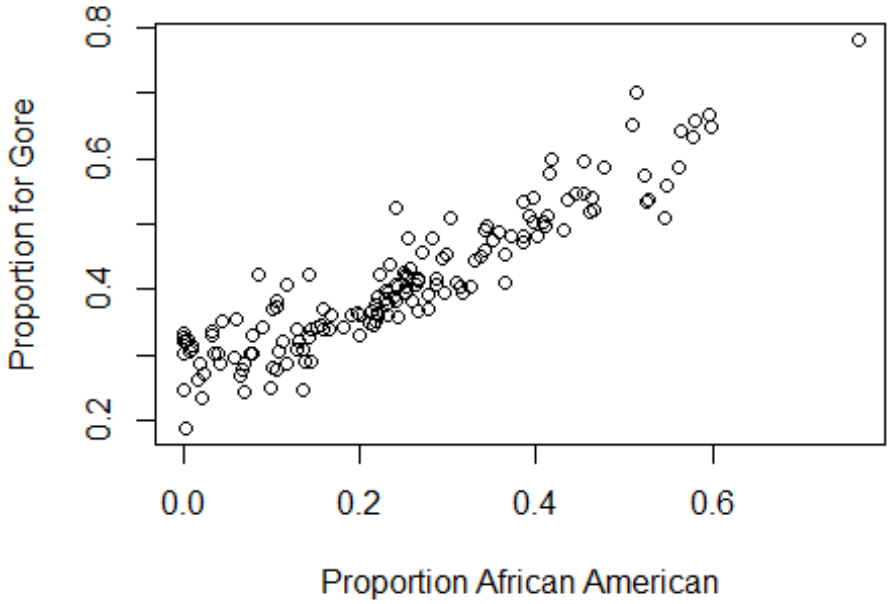
우선 gore 의 득표와 proportion of African American 의 관계에 대해 살펴보고자 한다.

그전에 단지 득표수만 고려한다면 인구 수에 따라 편차가 크기 때문에 득표율을 기준으로 하기로 한다.

Plot 명령어를 사용하고 알고 싶은 두 변수를 방정식의 형태로 argument 에 넣는다.

(y 축에 나타나는 변수 ~ x 축에 나타나는 변수)

```
gavote$pergore <- gavote$gore/gavote$votes
plot(pergore ~ perAA, gavote, xlab='Proportion African American', ylab='Propo
rtion for Gore')
```

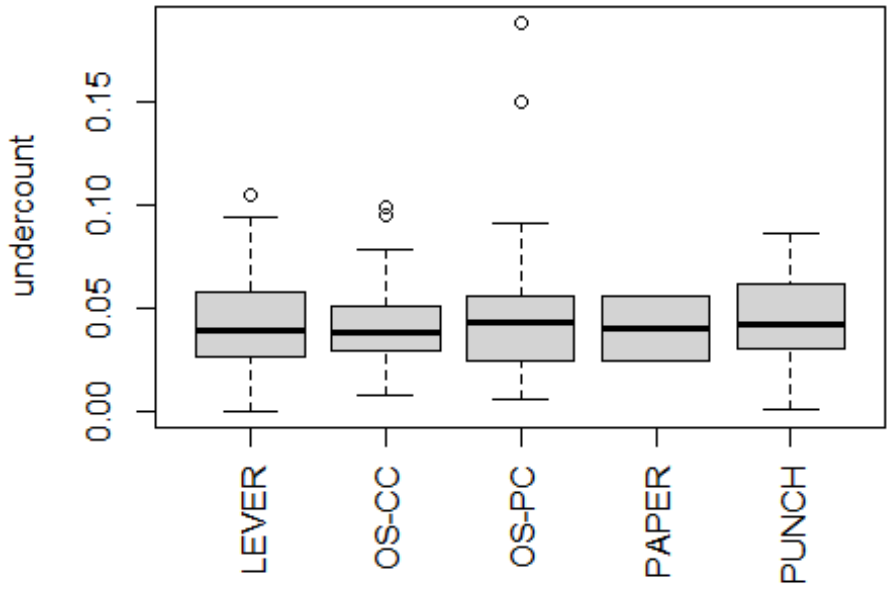


➔ 두 변수 간의 양의 상관관계가 있다는 것을 알 수 있다.

2-5. Side-by-side boxplots

만약 Quantitative 와 Qualitative 변수 간의 관계를 알고 싶다면 boxplot 을 이용하면 된다.

```
plot(undercount ~ equip, gavote, xlab='', las=3)
```



➔ Equip 차이에 따른 undercount 차이는 별로 없는 것으로 보인다.

2-6. Cross-tabulations

만약 level 이 세 네 개가 넘는 두 Qualitative 간의 관계를 알아보고 싶다면 cross-tabulation 을 이용하면 된다. Cross-tabulation 은 두 변수를 기준으로 하여 각 빈도수를 나타내는 표이다. 예를 들어 아래의 표에서 atlanta 에 해당하면서 rural 에 해당하는 county 는 한 곳뿐이라는 것을 알 수 있다. Cross-tabulation 을 보여줄 수 있는 명령어는 xtabs 이다.

```
xtabs(~ atlanta + rural, gavote)
```

```
##             rural
## atlanta      rural urban
##   Atlanta        1    14
##   notAtlanta   116    28
```

2-7. Change the variable name

그런데 rural 의 경우 level 들의 이름 중 하나와 factor 의 명이 같다는 문제점이 있다. 따라서 헷갈림을 방지하기 위해 우리는 rural 변수의 변수명을 바꿔주고자 한다.

Names 명령어를 통해 gavote dataset 의 변수명들을 확인할 수 있다.

```
names(gavote)
```

```
##  [1] "equip"      "econ"       "perAA"      "rural"      "atlanta"
##  [6] "gore"       "bush"       "other"      "votes"      "ballots"
## [11] "undercount" "pergore"
```

Rural 의 경우 변수 명 list 에서 4 번 째에 위치하므로 4 번 째 요소의 이름을 usage 로 바꿔주기로 한다.

```
names(gavote)[4] <- 'usage' ## 변수 내의 level 의 label 이랑 이름이 중복돼서 혼란을 피하기 위해 바꿔준다.
```

2-8. Correlation

Qualitative 변수들 간의 상관관계를 알아보기 위해서 Correlation 을 확인할 수도 있다.

Correlation 은 -1 에서 1 사이의 값으로 0 에서 멀어질수록 두 변수 간에 상관관계가 높다는 것을 알려준다(양의 값인 경우 양의 상관관계).

우리는 일단 perAA, ballots, undercount, pergore 간의 correlation 을 알아보고자 한다.

```
nix <- c(3,10,11,12) ## 해당 변수들의 위치 설정
```

correlation 을 계산해주는 명령어는 cor 이다.

```
cor(gavote[,nix])
```

```
##                perAA     ballots undercount    pergore
## perAA      1.0000000  0.02773230  0.2296874 0.92165247
## ballots    0.0277323  1.00000000 -0.1551724 0.09561688
## undercount 0.2296874 -0.15517245  1.0000000 0.21876519
## pergore    0.9216525  0.09561688  0.2187652 1.00000000
```

위의 결과로 알 수 있는 것은 앞서 scatter plot 에서도 알 수 있었듯이, African American 의 비율과 gore 의 득표율 간에는 강한 양의 상관관계가 있다는 것을 알 수 있다.

3. Fitting a Linear Model

이제 본격적으로 Linear Model 을 설정하고 분석을 진행할 것이다.

우리는 다음과 같은 linear model formula 를 상정하고 분석할 것이다.

$$\text{Undercount} = \beta_0(\text{intercept}) + \beta_1 \text{pergore} + \beta_2 \text{perAA} + \varepsilon$$

R 에서 Linear Model 식을 세워주는 명령어는 lm 이다. 그리고 '='을 '~'로 표현하고 변수들의 결합은 +로 표현한다.

```
lmod <- lm(undercount~pergore+perAA, gavote)
```

세워진 model 의 coefficient 값들(베타 추정치)을 확인해보면 다음과 같다.

```
coef(lmod)
```

```
## (Intercept)     pergore       perAA
##  0.03237600  0.01097872  0.02853314
```

3-2. Predicting

그리고 세워진 model 을 토대로 Response(undercount)의 값들의 추정치(fitted value)를 구해보면 다음과 같다.

```
predict(lmod)
```

```
##       APPLING      ATKINSON         BACON         BAKER       BALDWIN
##    0.04133661    0.04329088    0.03961823    0.05241202    0.04795484
##         BANKS       BARROW       BARTOW      BEN.HILL       BERRIEN
##    0.03601558    0.03794768    0.03824856    0.04568440    0.03951183
##          BIBB      BLECKLEY      BRANTLEY       BROOKS         BRYAN
##    0.04918830    0.04084040    0.03670210    0.04716214    0.03946337
##       BULLOCH         BURKE         BUTTS       CALHOUN        CAMDEN
##    0.04272642    0.05136792    0.04401044    0.05485224    0.04179928
##       CANDLER       CARROLL       CATOOSA      CHARLTON       CHATHAM
##    0.04344251    0.03976103    0.03602163    0.04292433    0.04750227
## CHATTAHOOCHEE     CHATTOOGA      CHEROKEE       CLARKE          CLAY
##    0.04764892    0.03942706    0.03555377    0.04504196    0.05549852
##       CLAYTON        CLINCH          COBB        COFFEE      COLQUITT
##    0.05405937    0.04421053    0.04099018    0.04394917    0.04171247
##      COLUMBIA          COOK        COWETA      CRAWFORD         CRISP
##    0.03793540    0.04412875    0.03966886    0.04419461    0.04611705
##          DADE        DAWSON       DECATUR        DEKALB         DODGE
##    0.03605864    0.03511850    0.04669830    0.05475324    0.04331231
##         DOOLY     DOUGHERTY       DOUGLAS         EARLY        ECHOLS
##    0.05151597    0.05358825    0.04115587    0.04775781    0.03789845
##     EFFINGHAM        ELBERT       EMANUEL         EVANS        FANNIN
##    0.03881787    0.04444902    0.04584928    0.04515983    0.03598472
##       FAYETTE         FLOYD       FORSYTH      FRANKLIN         FULTON
##    0.03833868    0.03963718    0.03451702    0.03800347    0.05058744
##        GILMER      GLASCOCK         GLYNN       GORDON         GRADY
##    0.03569792    0.03705342    0.04223991    0.03694448    0.04373769
##        GREENE      GWINNETT     HABERSHAM         HALL       HANCOCK
##    0.04733356    0.03915798    0.03575577    0.03718296    0.06279187
```

```
##      HARALSON         HARRIS           HART          HEARD          HENRY
##    0.03751672    0.04064671    0.04107793    0.03930191    0.03963794
##       HOUSTON         IRWIN       JACKSON        JASPER    JEFF.DAVIS
##    0.04200207    0.04290696    0.03728655    0.04390438    0.04004298
##     JEFFERSON       JENKINS       JOHNSON         JONES         LAMAR
##    0.05324856    0.04825427    0.04439236    0.04313121    0.04425755
##        LANIER       LAURENS           LEE       LIBERTY       LINCOLN
##    0.04385018    0.04507076    0.03892391    0.05067472    0.04573235
##          LONG       LOWNDES       LUMPKIN         MACON       MADISON
##    0.04339329    0.04517635    0.03613962    0.05581840    0.03751186
##        MARION      MCDUFFIE      MCINTOSH    MERIWETHER        MILLER
##    0.04692807    0.04574544    0.04924682    0.04919190    0.04265158
##      MITCHELL        MONROE    MONTGOMERY        MORGAN        MURRAY
##    0.04981344    0.04307184    0.04374217    0.04313021    0.03598129
##      MUSCOGEE        NEWTON        OCONEE    OGLETHORPE      PAULDING
##    0.04960889    0.04200194    0.03670443    0.04063029    0.03733922
##         PEACH       PICKENS        PIERCE          PIKE          POLK
##    0.04954864    0.03590777    0.03847804    0.03948920    0.04017706
##       PULASKI        PUTNAM       QUITMAN        RABUN      RANDOLPH
##    0.04455514    0.04454688    0.05088522    0.03608373    0.05331472
##      RICHMOND      ROCKDALE        SCHLEY       SCREVEN      SEMINOLE
##    0.05132390    0.04023190    0.04457407    0.04855631    0.04511893
##      SPALDING      STEPHENS       STEWART        SUMTER        TALBOT
##    0.04318359    0.03870448    0.05658610    0.05004936    0.05616301
##    TALIAFERRO      TATTNALL        TAYLOR       TELFAIR       TERRELL
##    0.05671849    0.04243780    0.04913822    0.04657823    0.05354144
##        THOMAS          TIFT        TOOMBS        TOWNS      TREUTLEN
##    0.04576222    0.04230964    0.04250660    0.03602581    0.04565230
##         TROUP        TURNER        TWIGGS         UNION         UPSON
##    0.04327165    0.04764528    0.05110591    0.03589563    0.04343877
##        WALKER        WALTON          WARE        WARREN    WASHINGTON
##    0.03703049    0.03889793    0.04265003    0.05416216    0.05126179
##         WAYNE       WEBSTER       WHEELER         WHITE     WHITFIELD
##    0.04046588    0.05186413    0.04486532    0.03605758    0.03683139
##        WILCOX        WILKES     WILKINSON         WORTH
##    0.04440304    0.04868008    0.04954797    0.04397619
```

3-3. Residuals

Residual($\hat{\varepsilon} = y - X\hat{\beta}$) 또한 구할 수 있다.

```
residuals(lmod)
```

```
##       APPLING      ATKINSON         BACON         BAKER       BALDWIN
##  3.694660e-02 -6.994927e-03  6.555058e-02  2.348407e-03  3.589940e-03
##         BANKS        BARROW       BARTOW      BEN.HILL        BERRIEN
##  1.426726e-02 -4.406713e-03  1.987376e-03  1.424361e-01 -2.498669e-02
##          BIBB      BLECKLEY      BRANTLEY        BROOKS          BRYAN
```

```
## -5.040438e-03 -1.032579e-02  3.335900e-02 -4.461253e-03 -3.946337e-02
##       BULLOCH         BURKE         BUTTS       CALHOUN        CAMDEN
## -4.407512e-03 -1.153479e-02  2.036622e-03  3.134631e-02 -3.346921e-02
##       CANDLER       CARROLL       CATOOSA      CHARLTON       CHATHAM
##  1.274103e-04 -2.401269e-03 -1.432571e-02  6.687554e-03 -2.317240e-02
## CHATTAHOOCHEE     CHATTOOGA      CHEROKEE        CLARKE          CLAY
## -3.134900e-02  1.128378e-02 -1.478981e-02 -2.420792e-02 -1.276988e-02
##       CLAYTON        CLINCH          COBB        COFFEE      COLQUITT
## -2.387409e-02 -1.064340e-02 -3.465258e-02 -1.741877e-02 -4.725510e-03
##      COLUMBIA          COOK        COWETA      CRAWFORD         CRISP
##  4.557823e-03  3.437592e-02 -1.228910e-02 -2.629814e-02  1.482385e-02
##          DADE        DAWSON       DECATUR        DEKALB         DODGE
##  9.499013e-03 -1.762724e-02 -1.504472e-02 -1.809054e-02 -1.568160e-02
##         DOOLY     DOUGHERTY       DOUGLAS         EARLY        ECHOLS
##  1.106725e-02  1.302490e-03 -1.047622e-02  8.324335e-03  2.080176e-02
##     EFFINGHAM        ELBERT       EMANUEL         EVANS        FANNIN
##  2.991246e-02 -2.649599e-02 -4.584928e-02 -6.758328e-03 -1.351190e-03
##       FAYETTE         FLOYD       FORSYTH      FRANKLIN        FULTON
## -3.092625e-02 -3.871541e-03  1.164519e-03  3.008646e-02  1.263527e-02
##        GILMER      GLASCOCK         GLYNN        GORDON         GRADY
## -3.397497e-03  3.567386e-02 -9.109594e-03  1.685814e-02  2.464313e-02
##        GREENE      GWINNETT     HABERSHAM          HALL       HANCOCK
##  1.627004e-02 -3.295423e-02  3.035196e-02 -9.231782e-03 -1.087175e-02
##      HARALSON        HARRIS          HART         HEARD         HENRY
##  1.382490e-02 -2.050480e-02 -1.585480e-02  1.145499e-02 -1.494596e-02
##       HOUSTON         IRWIN       JACKSON        JASPER    JEFF.DAVIS
## -1.182299e-02 -3.078365e-02 -8.488548e-03  1.414627e-02  3.713236e-02
##     JEFFERSON       JENKINS       JOHNSON         JONES         LAMAR
## -1.125200e-02 -7.203976e-03 -1.804991e-02 -1.893311e-04 -3.538550e-03
##        LANIER       LAURENS           LEE       LIBERTY       LINCOLN
##  1.497335e-02 -8.750812e-03 -8.665610e-03 -1.695379e-02  1.396462e-02
##          LONG       LOWNDES       LUMPKIN         MACON       MADISON
## -6.757149e-03 -2.669000e-02 -1.638940e-02 -1.940494e-03 -5.890828e-03
##        MARION      MCDUFFIE      MCINTOSH    MERIWETHER        MILLER
## -4.601315e-02 -4.469098e-03  2.483619e-02  1.614664e-02 -1.330621e-02
##      MITCHELL        MONROE    MONTGOMERY        MORGAN        MURRAY
##  2.854002e-02 -5.302281e-03 -1.886848e-02  4.780742e-03  6.343119e-03
##      MUSCOGEE        NEWTON        OCONEE    OGLETHORPE      PAULDING
## -7.237717e-03 -1.181356e-03 -1.971648e-02  6.523126e-03 -9.800841e-03
##         PEACH       PICKENS        PIERCE          PIKE          POLK
## -2.775587e-02 -2.362305e-02  5.630602e-03 -5.859127e-03 -1.774015e-03
##       PULASKI        PUTNAM       QUITMAN         RABUN      RANDOLPH
##  1.211247e-02  1.698470e-02 -2.188415e-02 -8.597412e-04  9.630461e-02
##      RICHMOND      ROCKDALE        SCHLEY       SCREVEN      SEMINOLE
##  3.943407e-03 -1.094021e-02 -2.539142e-02 -2.517055e-02  1.407009e-02
##      SPALDING      STEPHENS       STEWART        SUMTER        TALBOT
## -1.742084e-02 -7.995867e-04  2.633931e-03  9.863392e-03 -2.925028e-02
##    TALIAFERRO      TATTNALL        TAYLOR       TELFAIR       TERRELL
## -1.099874e-03  2.644187e-02  4.943506e-02  4.732579e-02 -2.916246e-03
```

```
##        THOMAS          TIFT        TOOMBS         TOWNS      TREUTLEN
##  1.051077e-02  1.659632e-03  1.709950e-02 -9.191787e-03  4.890489e-02
##         TROUP        TURNER        TWIGGS         UNION         UPSON
## -6.743701e-03  2.939342e-02  1.815259e-02  8.028909e-04  1.446136e-02
##        WALKER        WALTON          WARE        WARREN    WASHINGTON
##  1.587605e-03 -3.416505e-03  2.104733e-02 -1.777402e-02 -1.719221e-02
##         WAYNE       WEBSTER       WHEELER         WHITE     WHITFIELD
##  5.198114e-05 -8.706238e-03  4.630606e-02  4.144113e-03 -1.369061e-02
##        WILCOX        WILKES     WILKINSON         WORTH
## -1.765407e-02 -2.208434e-02 -3.583489e-02  1.749795e-02
```

3-4. Deviance or RSS

우리는 이제 우리의 model 이 우리의 데이터에 얼마나 잘 부합(fit)하는 지를 확인해보고자 한다.

Linear Model 의 경우 RSS($\hat{\varepsilon}^T\hat{\varepsilon}$)를 통해 이를 확인하는데 R 에서는 deviance 명령어를 통해 이 값을 확인할 수 있다(deviance 명령어는 RSS 보다 더 일반적인 measure 인데, linear model 의 경우 RSS 값을 출력해준다).

```
deviance(lmod)
```

```
## [1] 0.09324918
```

3-5. Degrees of freedom

자유도도 구해줄 수 있다. 자유도는 관측치의 수에서 coefficient 의 개수를 빼준 값으로 df.residual 명령어를 통해서 구할 수도 있고 그냥 직접 계산할 수도 있다.

```
df.residual(lmod)
```

```
## [1] 156
```

```
#number of cases - the number of coefficients
nrow(gavote) - length(coef(lmod))
```

```
## [1] 156
```

3-6. Estimation of standard error of error

Error 의 표준오차는 $\sqrt{RSS/df}$이므로 다음과 같이 계산할 수 있다.

```
sqrt(deviance(lmod)/df.residual(lmod))
```

```
## [1] 0.02444895
```

아니면 summary 명령어를 이용하여 다음과 같이 구할 수도 있다.

```
lmodsum <- summary(lmod)
lmodsum$sigma
```

```
## [1] 0.02444895
```

3-7.coefficient of determination or percentage of variance explained or R^2

deviance 의 경우 model 이 얼마나 fit 한 지를 절대적인 관점에서는 알려주지만, 상대적인 관점에서는 알려주지 않는다. 따라서 상대적인 관점에서 model 이 얼마나 fit 한지를 알려주는 coefficient of determination(또는 R^2) 값을 살펴보자.

Summary 에 내장된 것을 활용할 수도 있고

```
lmodsum$r.squared
```

```
## [1] 0.05308861
```

직접 계산을 통해서 구할 수도 있다.

```
cor(predict(lmod), gavote$undercount)^2
```

```
## [1] 0.05308861
```

3-8.Adjusted R^2

그런데 R^2 은 predictor 의 개수가 올라가면 덩달아 올라간다는 단점이 있다. 따라서 이를 보완해주기 위해서 adjusted R^2 을 사용한다.

Adjusted R^2 역시 summary 에 내장되어 있다.

```
lmodsum$adj.r.squared
```

```
## [1] 0.04094872
```

3-9.summary

Summary 명령어를 통해 여지껏 계산했던 것들을 한 번에 확인할 수 있다.

```
summary(lmod)
```

```
##
## Call:
## lm(formula = undercount ~ pergore + perAA, data = gavote)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.046013 -0.014995 -0.003539  0.011784  0.142436
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.03238    0.01276   2.537   0.0122 *
## pergore      0.01098    0.04692   0.234   0.8153
## perAA        0.02853    0.03074   0.928   0.3547
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02445 on 156 degrees of freedom
## Multiple R-squared:  0.05309,    Adjusted R-squared:  0.04095
## F-statistic: 4.373 on 2 and 156 DF,  p-value: 0.01419
```

만약 위의 요약이 너무 양이 많고 복잡해 보인다면, faraway package 안에 있는 sumary 함수를 이용해서 좀 더 간단하게 살펴볼 수 있다.

```
library(faraway)
sumary(lmod)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.032376   0.012761  2.5370  0.01216
## pergore     0.010979   0.046922  0.2340  0.81531
## perAA       0.028533   0.030738  0.9283  0.35470
##
## n = 159, p = 3, Residual SE = 0.02445, R-Squared = 0.05
```

4. Interpretation

이제 qualitative 변수들을 좀 더 추가하고 이를 해석해보도록 하자.

우선 좀 더 명확하게 하기 위해서 pergore 와 perAA 등 quantitative 변수들을 중앙화했다.

```
gavote$cpergore <- gavote$pergore - mean(gavote$pergore)
gavote$cperAA <- gavote$perAA - mean(gavote$perAA)
```

그리고 model 의 방정식에 usage 와 equip 이라는 qualitative 변수들을 추가하였으며 cpergore 와 usage 의 경우 interaction term 이 있는 것을 상정하였다.

```
lmodi <- lm(undercount ~ cperAA+cpergore*usage+equip, gavote)
```

이를 formula 로 표현하면

$$\begin{aligned} &undercount \\ &= \beta_0(\text{intercept}) + \beta_1 \text{cperAA} + \beta_2 \text{cpergore} + \beta_3 \text{usageurban} + \beta_4 \text{equipOSCC} + \beta_5 \text{equipOSPC} \\ &+ \beta_6 \text{equipPAPER} + \beta_7 \text{equipPUNCH} + \beta_8 \text{cpergore: usageurban} + \varepsilon \end{aligned}$$

세워진 model 의 관련 값들을 살펴보면

```
summary(lmodi)
```

```
                      Estimate Std. Error t value  Pr(>|t|)
(Intercept)          0.0432973  0.0028386 15.2529 < 2.2e-16
cperAA               0.0282641  0.0310921  0.9090  0.364786
cpergore             0.0082368  0.0511562  0.1610  0.872299
usageurban          -0.0186366  0.0046482 -4.0095 9.564e-05
equipOS-CC           0.0064825  0.0046799  1.3852  0.168060
equipOS-PC           0.0156396  0.0058274  2.6838  0.008097
equipPAPER          -0.0090920  0.0169263 -0.5372  0.591957
equipPUNCH           0.0141496  0.0067827  2.0861  0.038658
cpergore:usageurban -0.0087995  0.0387162 -0.2273  0.820515


n = 159, p = 9, Residual SE = 0.02335, R-Squared = 0.17
```

해석:

1. usageurban, equipOSCC, equipOSPC, equipPAPER, equipPUNCH 는 모두 dummy variable 이다.

2. cpergore:usageurban 은 usageurban dummy variable 과 cpergore value 값의 곱이다. 즉, usagerural 이라면 0 값이 나온다.

3. 평균적인 gore voter 와 african american 들을 가지고 있고 lever machine 을 사용한 rural county 를 고려해보자. 이 때 rural 과 lever 는 위에서 reference level 로 사용되었다. 따라서 predicted undercount 에 이러한 term 들은 기여하지 않는다. 또한 두 quantitative 변수들을 모두 centered 해줬기 때문에 이 term 들도 prediction 에 들어가지 않는다. 만약 이러한 term 들이 모두 빠진다면 predicted undercount 는 intercept 의 추정치인 4.33%가 된다.

4. 만약 equipment 로 OS-PC 를 사용하는 것을 제외하고 다른 변수들은 변하지 않는다면, predicted undercount 는 1.56% 상승할 것이다. 다만 조심할 것은 lever machine 을 사용하는 지역에 비해 OS-PC 를 사용하는 지역의 undercount 가 1.56% 높다고 생각하면 안 된다. 다른 변수들의 영향도 있기 때문이다. 하지만 만약 lever machine 을 사용하는 지역이 만약 OS-PC 로 갈아탄다면 1.56% 상승할 것이라고는 얘기할 수 있다.

5. 만약 African American 이 아예 없는 지역에서 모두 African American 인 지역으로 county 가 바뀐다면 undercount 는 2.83% 증가할 것이다.

6. usage 와 pergore 의 해석은 따로따로 이뤄질 수 없다. 왜냐하면 interaction term 이 존재하기 때문이다. 시골지역에서는 gore voters 가 10% 증가한다면 undercount 가 0.08% 증가할 것이다. 그러나 도시 지역에서는 (0.00824-0.00880)*10=-0.0056% 로 오히려 감소할 것이라고 예측해야 한다. 즉 pergore 에서 interaction estimation 을 더해주어야 한다.

5. Hypothesis Testing

만약 두 모델 중 어떤 모델이 더 적합한 지를 알고 싶다면 F-test 를 사용하면 된다. R 에서는 anova 를 이용해서 F-Test 를 진행할 수 있는데 만약 p-value 가 0.05 보다 낮다면 Model 1 을 기각한다. 즉, model 2 를 채택한다.

```
anova(lmod, lmodi)
```

```
## Analysis of Variance Table
##
## Model 1: undercount ~ pergore + perAA
## Model 2: undercount ~ cperAA + cpergore * usage + equip
##   Res.Df      RSS Df Sum of Sq      F   Pr(>F)
## 1    156 0.093249
## 2    150 0.081775  6  0.011474 3.5077 0.002823 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

각각의 predictors 들에 대해서도 test 를 진행할 수 있다. drop1 명령어를 활용하면 우리의 full model 과 full model 에서 변수들을 하나씩 제외했을 때의 model 을 F-test 해준다. 따라서 만약 p-value 값이 낮다면 그 변수가 중요하다는 것을 알려준다.

```
drop1(lmodi, test='F')
```

```
## Single term deletions
##
## Model:
## undercount ~ cperAA + cpergore * usage + equip
##                Df Sum of Sq      RSS     AIC F value  Pr(>F)
## <none>                      0.081775 -1186.1
## cperAA          1 0.0004505 0.082226 -1187.2  0.8264 0.36479
## equip           4 0.0054438 0.087219 -1183.8  2.4964 0.04521 *
## cpergore:usage  1 0.0000282 0.081804 -1188.0  0.0517 0.82051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

그런데 여기서 알 수 있는 것은 cpergore:usage 에서 interaction term 만 고려되고 main effect 인 cpergore 와 usage 는 고려되지 않았다는 것이다.

이는 Hierarchy principle 로, interaction 에 대응되는 lower-order term 들은 model 안에 유지되어야 한다는 원칙에 따른 것이다.

따라서 여기서는 interaction 은 유의미하지 않다는 것을 알지만 main effect 들에 살펴보기 위해서는 추가적인 step 이 필요하다.

6. Confidence Interval

각 predictor 들의 Confidence Interval 을 구할 수 있는데 여기서 만약 C.I 가 0 을 포함하는 구간이라면 해당 변수의 t-test 의 p-value 값은 5%이상이다. 따라서 이를 통해 어떤 변수가 통계적으로 유의한 지를 확인해볼 수 있다.

```
confint(lmodi)
```

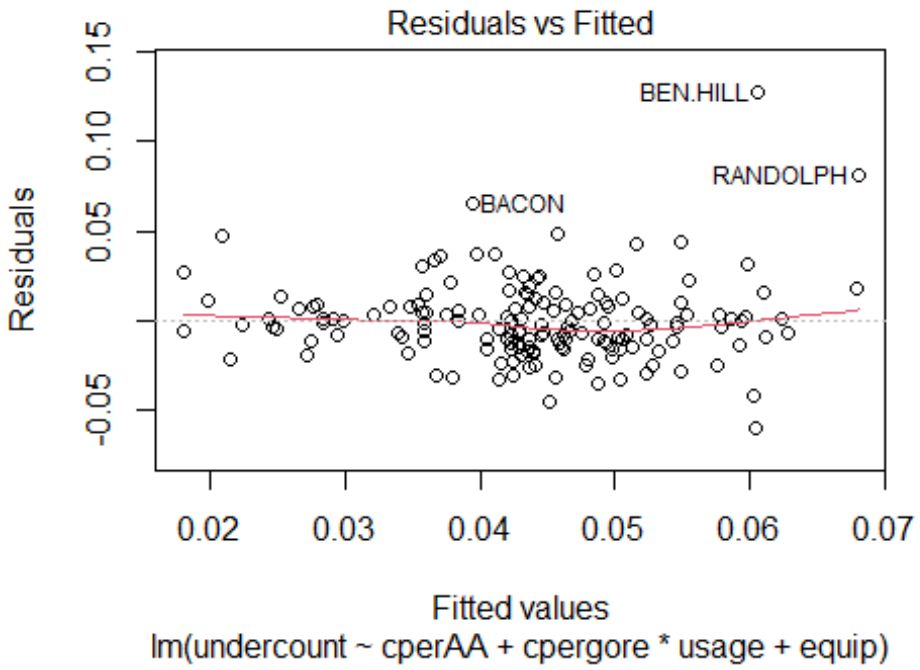
```
##                              2.5 %       97.5 %
## (Intercept)          0.0376884415  0.048906189
## cperAA              -0.0331710614  0.089699222
## cpergore            -0.0928429315  0.109316616
## usageurban          -0.0278208965 -0.009452268
## equipOS-CC          -0.0027646444  0.015729555
## equipOS-PC           0.0041252334  0.027153973
## equipPAPER          -0.0425368415  0.024352767
## equipPUNCH           0.0007477196  0.027551488
## cpergore:usageurban -0.0852990903  0.067700182
```

7. Diagnostics

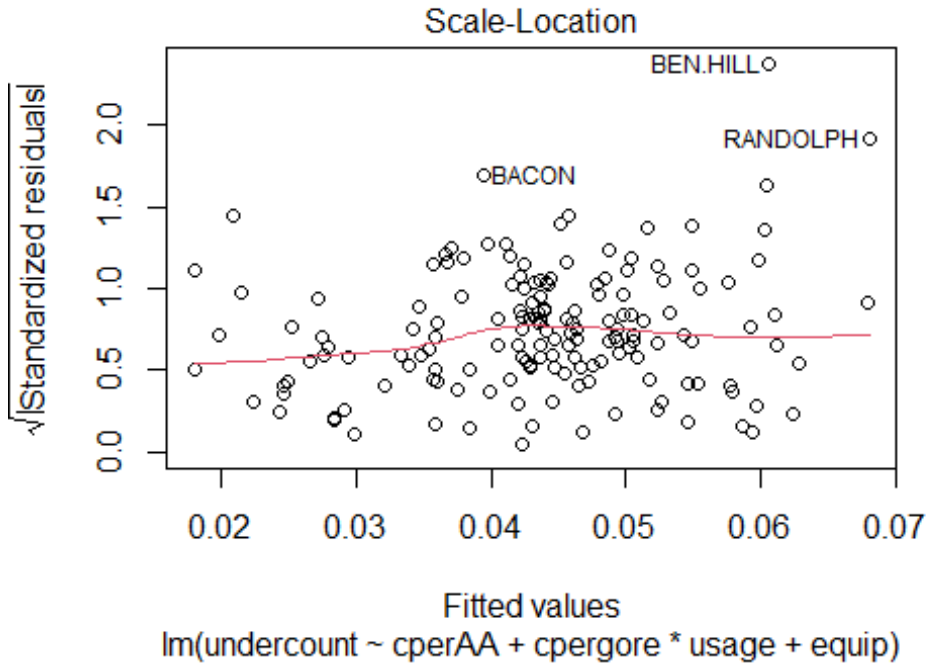
우리는 Linear Model 을 세울 때 여러 가정들을 상정한다. 여기서는 그 가정들이 유효한 지를 살펴볼 것이다.

Plot 명령어는 우리 Model 의 가정 점검에 대한 유용한 4 개의 graph 를 출력해준다.

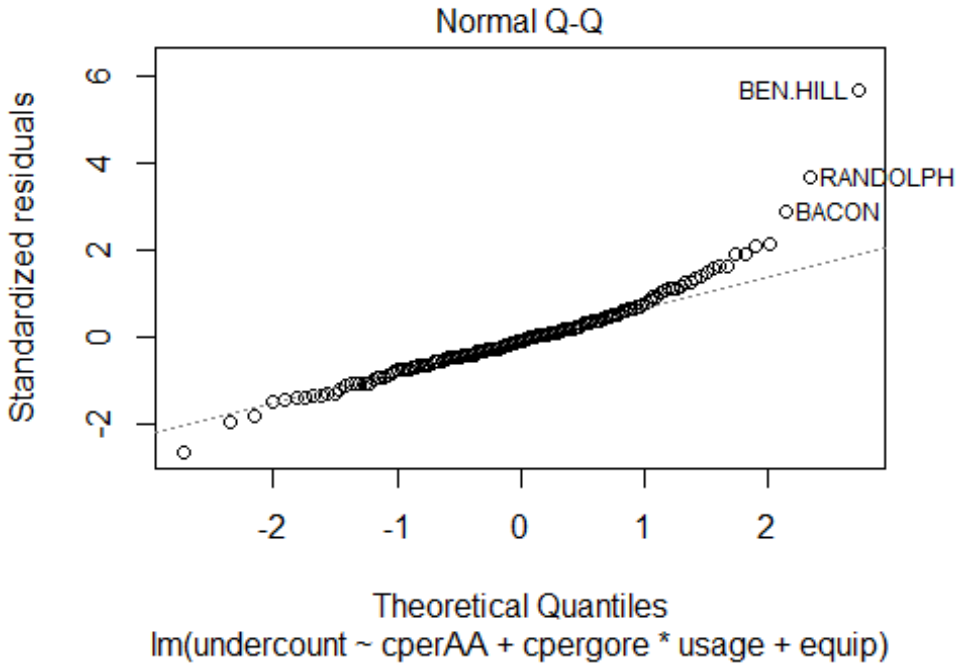
```
plot(lmodi)
```



➔ 이 plot 은 lack of fit 를 감지할 수 있게 해준다. 만약 residual 이 curvilinear trend 를 보여준다면, 이는 어느 한 변수의 변형 등, model 에 수정이 필요하다는 것을 의미한다. 또한 이 plot 은 constant variance of error assumption 을 체크하는 데에도 사용될 수 있다. 여기에서는 outlier 들이 몇 개 존재하지만 전체적으로 봤을 때 우리의 가정에 큰 문제는 없는 것으로 보인다.



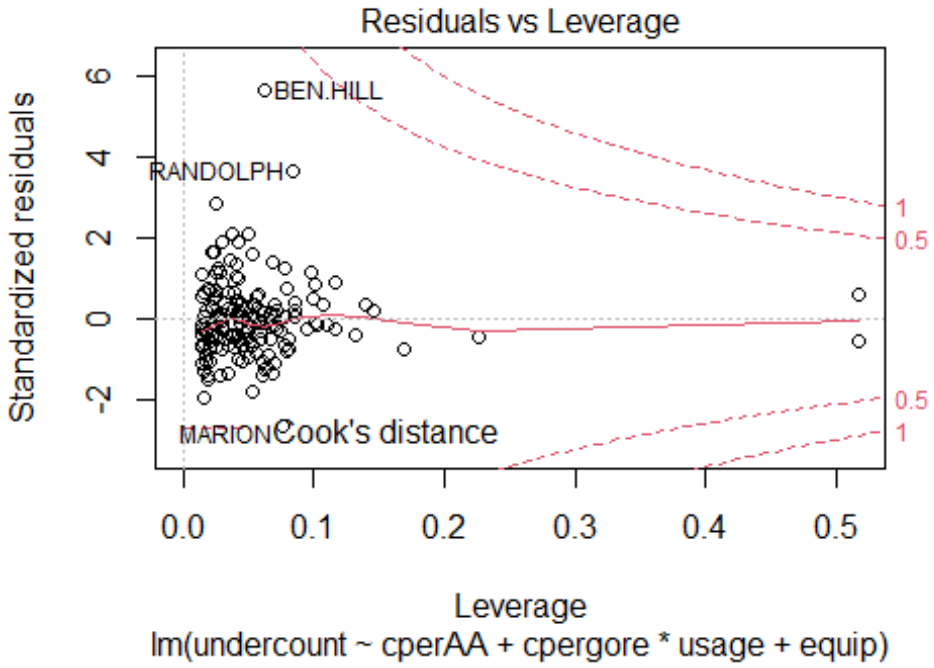
➔ 여기서는 Residual 에 square root 를 씌워준 것에 대한 plot 이다.



➔ 이 plot 은 sorted residual 과 $\Phi^{-1}\left(\frac{i}{n+1}\right) for\ i = 1, \dots, n$을 비교하는 plot 이다.

➔ 만약 이 plot 에서 point 들이 점선을 따라 위치한다면 normality of error 가정은 합리적이라고 할 수 있다.

➔ 다만 끝 부분에서 차이가 벌어지는 것을 알 수 있는데, 이는 우리가 robust fitting method 를 사용하는 것을 고려해야 함을 보여준다. Robust fitting 은 뒤에서 설명한다.



➔ Leverage 는 hat matrix 의 대각행렬 값이다. 그리고 Leverage 값이 높으면 $var(\hat{\varepsilon})$값이 작아지기 때문에 높은 leverage 값은 어떤 case 가 influential 이 될 가능성이 높은 지를 알려준다.

➔ Cook's Distance 는 influence 를 확인하기에 유명한 방법이다. Cook's Distance 의 값이 높을수록 influential 한 case 이다.

➔ 위의 plot 에서는 두 개의 case 가 비교적 높은 cook's distance 를 가지고 있다는 것을 알 수 있다.

Cooks.distance 명령어를 통해 cooks.distance 가 일정 수준을 넘는 case 들을 찾아낼 수 있다. 여기서는 0.1 을 기준으로 한다.

```
gavote[cooks.distance(lmodi)>0.1,]
```

```
##           equip econ perAA usage    atlanta gore bush other votes ballots
## BEN.HILL OS-PC poor 0.282 rural notAtlanta 2234 2381    46  4661    5741
## RANDOLPH OS-PC poor 0.527 rural notAtlanta 1381 1174    14  2569    3021
##           undercount   pergore   cpergore     cperAA
```

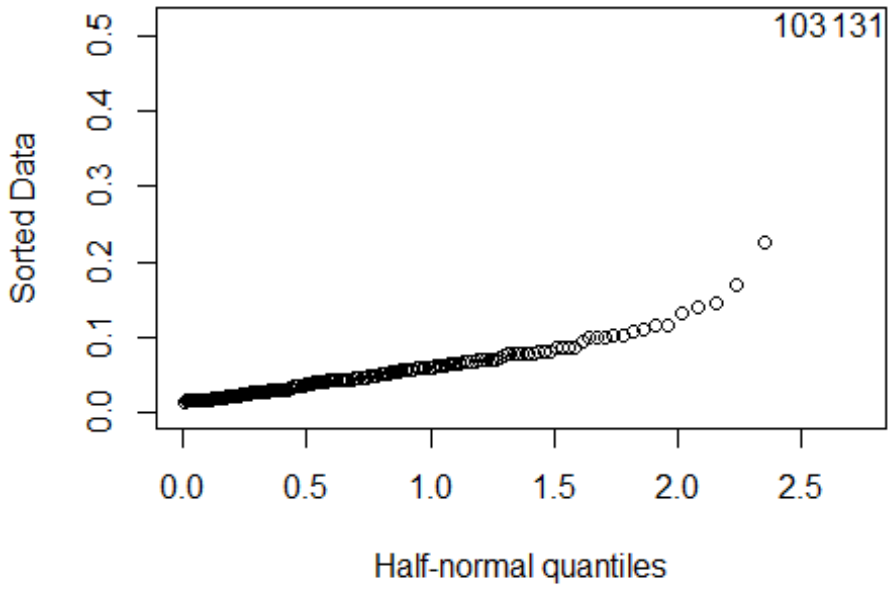
```
## BEN.HILL  0.1881205 0.4792963 0.07097452 0.03901887
## RANDOLPH  0.1496193 0.5375633 0.12924148 0.28401887
```

여기서의 case 들은 기대값보다 훨씬 높은 undercounts 값들을 가지고 있다는 것을 알 수 있다. 따라서 이 case 들은 influential 이다.

일련의 양의 관측치에서 일부 사례가 비정상적으로 극단적인지 여부를 판단하는 데 유용한 기법은 half-normal plot 이다.

여기서는 선형성을 보는 대신 outlier 에 집중하면 된다.

```
halfnorm(hatvalues(lmodi))
```



Hatvalue 가 높은 case 들을 찾아보자.

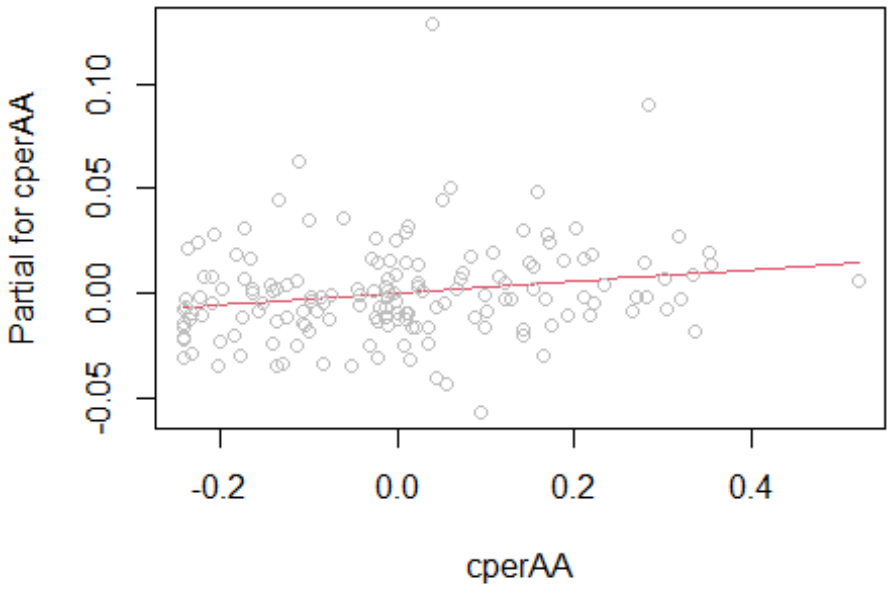
```
gavote[hatvalues(lmodi)>0.3,]
```

```
##            equip econ perAA usage    atlanta gore bush other votes ballots
## MONTGOMERY PAPER poor 0.243 rural notAtlanta 1013 1465    31  2509    2573
## TALIAFERRO PAPER poor 0.596 rural notAtlanta  556  271     5   832     881
##            undercount   pergore     cpergore       cperAA
## MONTGOMERY 0.02487369 0.4037465 -0.004575261 1.886792e-05
## TALIAFERRO 0.05561862 0.6682692  0.259947458 3.530189e-01
```

위의 case 들은 High leverage 를 갖지만 predictor 의 값들이 remarkable 하지 않다는 점에서 influential 하지는 않다.

Partial residual plot 은 x_i대 $\hat{\varepsilon} + \hat{\beta}_i x_i$의 관계를 보여준다. 즉, 다른 variable 을 제외하고 Response 와 해당 predictor 간의 관계를 보여준다.

```
termplot(lmodi, partial=TRUE, terms=1)
```



➔ cperAA 와 Response 는 양의 상관관계를 나타낸다.

8. Robust Regression

Least Square 방법은 normal error 에 대해서는 잘 작동하지만, long-tailed error 에 대해서는 형편없는 성능을 보여준다. 따라서 이를 보완하기 위해 outlier 들을 제거해줄 수도 있지만 단순히 outlier 를 제거하는 것은 또다른 outlier 들을 만들어내는 등의 문제를 야기할 수 있다. 따라서 robust regression 을 이용하는 것이 하나의 방안이 될 수 있다.

우선 흔히 사용되는 방법은 larger error 에 대한 비중을 낮추는 것이다. 이를 Huber method 라고 하는데, MASS 패키지의 rlm 명령어가 이 방식을 기본적으로 사용한다.

```r
library(MASS)
rlmodi<- rlm(undercount ~ cperAA+cpergore*usage+equip, gavote)
summary(rlmodi)
```

```
##
## Call: rlm(formula = undercount ~ cperAA + cpergore * usage + equip,
##     data = gavote)
## Residuals:
##        Min         1Q     Median         3Q        Max
## -6.026e-02 -1.165e-02 -6.587e-06  1.100e-02  1.379e-01
##
## Coefficients:
##                      Value   Std. Error t value
## (Intercept)           0.0414  0.0023    17.8662
## cperAA                0.0327  0.0254     1.2897
## cpergore             -0.0082  0.0418    -0.1972
## usageurban           -0.0167  0.0038    -4.4063
## equipOS-CC            0.0069  0.0038     1.8019
## equipOS-PC            0.0081  0.0048     1.6949
## equipPAPER           -0.0059  0.0138    -0.4269
## equipPUNCH            0.0170  0.0055     3.0720
## cpergore:usageurban   0.0073  0.0316     0.2298
##
## Residual standard error: 0.01722 on 150 degrees of freedom
```

Robust Estimation method 를 사용하면 Inferential method 는 적용하기 어려워진다. 따라서 앞서 Least Square 를 사용했을 때의 summary 와 다른 모양을 확인할 수 있다. 앞서 Least Square method model 과의 가장 큰 차이점은 equip OS-PC 의 coefficient 가 절반 정도로 감소했단 것이다. 나머지 변수에서는 큰 차이점은 없다.

9. Weighted Least Squares

우리는 인구수가 더 적은 곳에서는 ballot 과 votes 의 숫자 자체가 작기 때문에 조금만 값이 변해도 undercount 의 값이 더 쉽게 변동할 수 있다는 것을 예상할 수 있다. 따라서 이를 보완해주기 위해서 각 case 들에 다른 weight 값을 부여해줄 수 있다. Weight 값은 보통 var(y_i)의 역수를 기준으로 하는데, 여기서는 ballot 의 역수를 기준으로 하기로 한다.

R 에서는 간단하게 lm 명령어에 weight argument 를 추가해주면 자동으로 weighted least square method 로 변환된다.

```r
wlmodi <- lm(undercount ~ cperAA+cpergore*usage+equip, gavote,
weights=ballots)
summary(wlmodi)
```

```
##
## Call:
## lm(formula = undercount ~ cperAA + cpergore * usage + equip,
##     data = gavote, weights = ballots)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1949 -0.9342 -0.0455  1.0443 11.4357
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.043687   0.003443  12.689  < 2e-16 ***
## cperAA                0.068093   0.027538   2.473  0.01453 *
## cpergore             -0.046885   0.052747  -0.889  0.37550
## usageurban           -0.017916   0.003721  -4.815 3.56e-06 ***
## equipOS-CC            0.005582   0.004648   1.201  0.23168
## equipOS-PC           -0.005823   0.004689  -1.242  0.21623
## equipPAPER           -0.014154   0.037294  -0.380  0.70483
## equipPUNCH            0.015661   0.005376   2.913  0.00413 **
## cpergore:usageurban   0.011998   0.035635   0.337  0.73683
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.179 on 150 degrees of freedom
## Multiple R-squared:  0.4099, Adjusted R-squared:  0.3784
## F-statistic: 13.02 on 8 and 150 DF,  p-value: 3.669e-14
```

결과를 확인해보면 Unweighted fit 과 매우 다르다는 것을 알 수 있다. 이는 결과가 몇 개의 large county 에 영향을 많이 받았기 때문이다.

그런데 우리는 Undercount 의 variation 이 단지 ballot 의 수에만 영향을 받는 가를 고려해야 한다. Undercount 에 영향을 주는 변수 중에 ballot size 와는 무관한 변수들도 존재한다. 따라서 우리는 이 effect 들의 상대적인 크기를 고려해보아야 한다.

가장 작은 county 에서 평균적인 undercount 를 고려했을 때 binomial 을 이용한 standard deviation 은

```
> sqrt(0.035*(1-0.035)/881)

[1] 0.006191697
```

##881 은 min(ballot)이고 0.035 는 overall average undercount 값.

그런데 이는 Unweighted method 에서의 standard error 값인 0.0233 보다 훨씬 작은 값이다. 그리고 이 effect 는 크기가 큰 다른 county 에서 훨씬 더 작게 나타날 것이다. 따라서 다른 변수들

이 variation 에 더 큰 영향을 미칠 것이기 때문에 weighted 로 하지 말고 그냥 unweighted 로 놔두는 것을 추천한다.

10. Transformation

변수들을 변환하는 것에 대해 살펴보자. 우선 Response 의 경우 skewed 되어있을 때(box-cox method 를 통해 확인 가능) MASS 패키지에서 boxcox function 을 이용하면 정규분포 형태로 바꿔주는 것이 가능하다. 그러나 Response 의 형태를 변환하는 것은 해석의 어려움 등 여러 문제를 가지고 있기 때문에 주의를 요해서 바꿔야 한다.

반면 Predictor 들을 변환하는 것은 덜 문제적이다. cperAA 의 degree 를 늘려서 계산해보자.

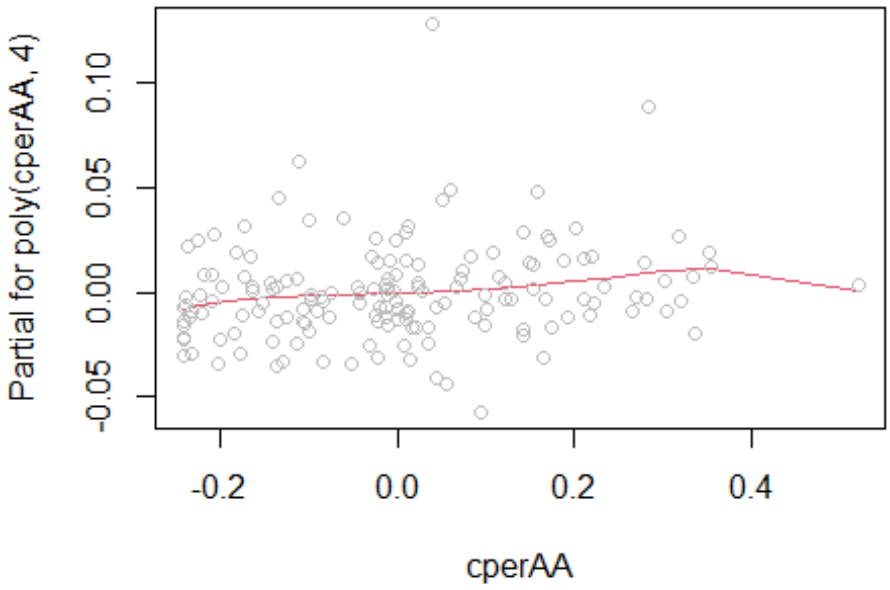
```
plmodi <- lm(undercount ~ poly(cperAA, 4) + cpergore*usage + equip, gavote)
summary(plmodi)
```

```
##
## Call:
## lm(formula = undercount ~ poly(cperAA, 4) + cpergore * usage +
##     equip, data = gavote)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.058563 -0.012963 -0.001987  0.009230  0.127984
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.043460   0.002875  15.115  < 2e-16 ***
## poly(cperAA, 4)1      0.052258   0.069391   0.753  0.45260
## poly(cperAA, 4)2     -0.002988   0.026135  -0.114  0.90914
## poly(cperAA, 4)3     -0.005363   0.024267  -0.221  0.82538
## poly(cperAA, 4)4     -0.016513   0.024199  -0.682  0.49606
## cpergore              0.013153   0.056930   0.231  0.81761
## usageurban           -0.019129   0.004741  -4.035 8.76e-05 ***
## equipOS-CC            0.006440   0.004720   1.364  0.17455
## equipOS-PC            0.015587   0.005879   2.652  0.00889 **
## equipPAPER           -0.010272   0.017204  -0.597  0.55137
## equipPUNCH            0.014053   0.006866   2.047  0.04247 *
## cpergore:usageurban -0.010538   0.041362  -0.255  0.79926
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02354 on 147 degrees of freedom
```

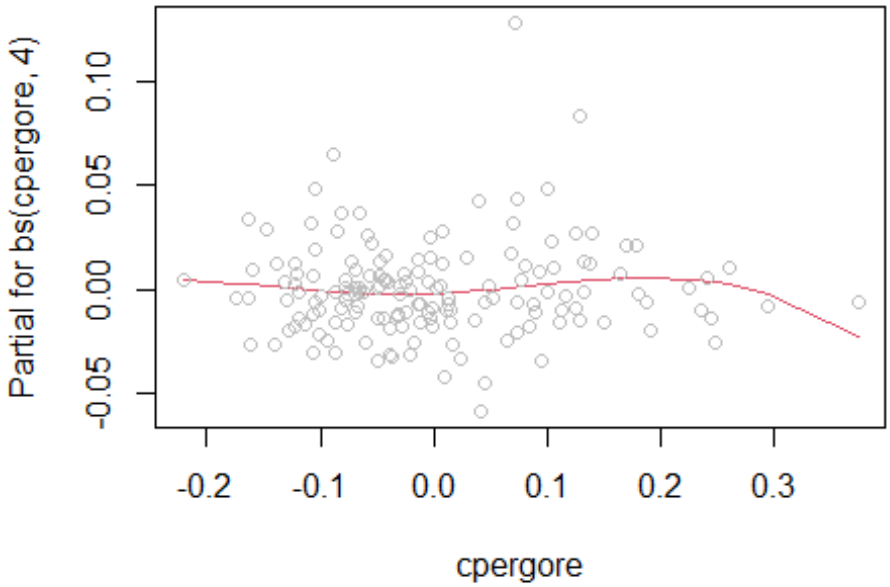
```
## Multiple R-squared:  0.1726, Adjusted R-squared:  0.1107
## F-statistic: 2.788 on 11 and 147 DF,  p-value: 0.002539
```

차수를 늘려보았지만 cperAA term 들은 모두 통계적으로 유의하지 않기 때문에 제거 가능하다.

```
termplot(plmodi, partial=TRUE, terms=1)
```




```
library(splines)
blmodi <- lm(undercount ~ cperAA + bs(cpergore, 4) + usage + equip, gavote)
termplot(blmodi, partial=TRUE, terms=2)
```



➔ Spline 과 Polynomial 을 비교해보면, Polynomial 의 경우 좀 더 진동이 큰 fit 을 보여주며 원래의 관찰 범위를 넘어서는 경우 예측 성능이 떨어진다. 그러나 spline 은 조금 더 stable 한 형태를 보여주며 더 좋은 local fit 을 보여준다. 따라서 extrapolation properties 도 좀 더 안정적이다. (성능이 더 좋다)

11. Variable Selection

우리는 Response 를 예측함에 있어서 모든 Variable 이 필요하지는 않다. 오히려 적절한 Variable 을 고름으로써 더 좋은 예측 성능을 낼 수 있다. 따라서 이번에는 어떤 Variable 을 선택해야하는지에 대한 것을 살펴보고자 한다.

우선 R^2 을 기준으로 선택할 수 있다. 즉, R^2 이 높은 model 을 선택하는 것이다. 그러나 이는 오직 quantitative predictors 에만 적용할 수 있다는 문제점이 있다.

또다른 변수를 선택하는 기준에 있어서 Akaike Information Criterion(AIC)이라는 것이 있다.

AIC = -2 maximum log likelihood + 2p (p 는 parameter 의 개수)

AIC 가 작을수록 좋은 모델인 것이다.

R 에서는 step 명령어를 이용하면 AIC 가 제일 작은 모델을 골라준다.

```r
biglm <- lm(undercount ~ (equip+econ+usage+atlanta)^2 + (equip+econ+usage+atl
anta)*(perAA+pergore), gavote)
smallm <- step(biglm, trace=FALSE)
summary(smallm)
```

```
##
## Call:
## lm(formula = undercount ~ equip + econ + usage + perAA + equip:econ +
##     equip:perAA + usage:perAA, data = gavote)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.059044 -0.009856  0.000000  0.008238  0.082598
##
## Coefficients: (2 not defined because of singularities)
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.0435310  0.0051439   8.463 3.27e-14 ***
## equipOS-CC           -0.0128784  0.0072732  -1.771 0.078809 .
## equipOS-PC            0.0034922  0.0111518   0.313 0.754635
## equipPAPER           -0.0578329  0.0363659  -1.590 0.114038
## equipPUNCH           -0.0142618  0.0187785  -0.759 0.448854
## econpoor              0.0180113  0.0055429   3.249 0.001450 **
## econrich             -0.0157358  0.0124077  -1.268 0.206836
## usageurban           -0.0006736  0.0072367  -0.093 0.925971
## perAA                -0.0389879  0.0164453  -2.371 0.019124 *
## equipOS-CC:econpoor -0.0114503  0.0097147  -1.179 0.240550
## equipOS-PC:econpoor  0.0424178  0.0137504   3.085 0.002458 **
## equipPAPER:econpoor         NA         NA      NA       NA
## equipPUNCH:econpoor -0.0160832  0.0131594  -1.222 0.223704
## equipOS-CC:econrich  0.0047127  0.0151356   0.311 0.755987
## equipOS-PC:econrich -0.0111987  0.0167837  -0.667 0.505728
## equipPAPER:econrich         NA         NA      NA       NA
## equipPUNCH:econrich  0.0168340  0.0216487   0.778 0.438128
## equipOS-CC:perAA     0.1181524  0.0326074   3.623 0.000407 ***
## equipOS-PC:perAA     0.0321434  0.0451861   0.711 0.478056
## equipPAPER:perAA     0.1260840  0.0803911   1.568 0.119066
## equipPUNCH:perAA     0.1243346  0.0500734   2.483 0.014216 *
## usageurban:perAA    -0.0472147  0.0290180  -1.627 0.105984
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01964 on 139 degrees of freedom
## Multiple R-squared:  0.4554, Adjusted R-squared:  0.381
## F-statistic: 6.118 on 19 and 139 DF,  p-value: 4.37e-11
```

그런데 단순히 예측하는 것이 아니라 변수들 간의 관계를 설명하는 것에 관심이 있다면, 좀 더 manual 하게 변수들을 고를 수 있다. 이 때는 앞서 살펴본 F-test 가 사용될 수 있다.

```
drop1(smallm, test='F')
```

```
## Single term deletions
##
## Model:
## undercount ~ equip + econ + usage + perAA + equip:econ + equip:perAA +
##     usage:perAA
##             Df Sum of Sq      RSS     AIC F value   Pr(>F)
## <none>                   0.053627 -1231.1
## equip:econ   6 0.0075232 0.061150 -1222.3  3.2500 0.005084 **
## equip:perAA  4 0.0068439 0.060471 -1220.0  4.4348 0.002101 **
## usage:perAA  1 0.0010214 0.054649 -1230.1  2.6474 0.105984
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

보면 usage:perAA 의 p-value 가 높으므로 제거 가능하다.

이를 고려해서 최종 모델을 세우면 다음과 같다.

```
finalm <- lm(undercount ~ equip + econ + perAA + equip*econ + equip:perAA, ga
vote)
sumary(finalm)
```

```
##
## Coefficients: (2 not defined because of singularities)
##                        Estimate Std. Error t value  Pr(>|t|)
## (Intercept)           0.0418709  0.0050276  8.3282 6.497e-14
## equipOS-CC           -0.0113268  0.0073726 -1.5363 0.1266999
## equipOS-PC            0.0085750  0.0111781  0.7671 0.4442883
## equipPAPER           -0.0584275  0.0370141 -1.5785 0.1166874
## equipPUNCH           -0.0157513  0.0187454 -0.8403 0.4021751
## econpoor              0.0202659  0.0055290  3.6654 0.0003489
## econrich             -0.0169664  0.0123919 -1.3692 0.1731284
## perAA                -0.0420403  0.0165935 -2.5335 0.0123853
## equipOS-CC:econpoor -0.0109645  0.0098849 -1.1092 0.2692236
## equipOS-PC:econpoor  0.0483848  0.0137954  3.5073 0.0006076
## equipPUNCH:econpoor -0.0035601  0.0124266 -0.2865 0.7749211
## equipOS-CC:econrich  0.0022777  0.0153780  0.1481 0.8824646
## equipOS-PC:econrich -0.0133182  0.0170541 -0.7809 0.4361491
## equipPUNCH:econrich  0.0200315  0.0219974  0.9106 0.3640450
## equipOS-CC:perAA     0.1072494  0.0328551  3.2643 0.0013771
## equipOS-PC:perAA    -0.0059062  0.0434140 -0.1360 0.8919805
## equipPAPER:perAA     0.1291364  0.0818061  1.5786 0.1166763
## equipPUNCH:perAA     0.0868490  0.0464997  1.8677 0.0638751
##
## n = 159, p = 18, Residual SE = 0.02000, R-Squared = 0.43
```

12. conclusion

마지막으로 우리의 최종 모델을 해석해보자.

우선 interaction 을 해석하기 위해 table 을 만들어보자.

우선 econ 과 equip 간의 상호작용을 살펴보기 위해 perAA 는 0.233 으로 고정한 table 을 만든다.

```
pdf <- data.frame(econ=rep(levels(gavote$econ), 5), equip=rep(levels(gavote$e
quip), rep(3,5)), perAA=0.233)
pdf
```

```
##      econ equip perAA
## 1  middle LEVER 0.233
## 2    poor LEVER 0.233
## 3    rich LEVER 0.233
## 4  middle OS-CC 0.233
## 5    poor OS-CC 0.233
## 6    rich OS-CC 0.233
## 7  middle OS-PC 0.233
## 8    poor OS-PC 0.233
## 9    rich OS-PC 0.233
## 10 middle PAPER 0.233
## 11   poor PAPER 0.233
## 12   rich PAPER 0.233
## 13 middle PUNCH 0.233
## 14   poor PUNCH 0.233
## 15   rich PUNCH 0.233
```

그 다음 필요한 변수들은 모두 주어졌으므로 이에 해당하는 Expected Undercount 값을 계산한다.

```
pp <- predict(finalm, new=pdf)
```

```
xtabs(round(pp,3) ~ econ + equip, pdf)
```

```
##         equip
## econ      LEVER  OS-CC  OS-PC  PAPER  PUNCH
##   middle  0.032  0.046  0.039  0.004  0.037
##   poor    0.052  0.055  0.108  0.024  0.053
##   rich    0.015  0.031  0.009 -0.013  0.040
```

보면 econ 의 status 가 낮을수록 undercount 값이 높아진다는 것을 알 수 있다.

또한 OS-PC 의 경우 Poor 계층에서 확연한 문제를 보인다는 것을 알 수 있다.

같은 방식으로 perAA 와 equip 을 비교할 수 있다. 여기서는 perAA 를 임의로 세 개의 그룹으로 나눠줄 수 있다.

```
pdf <- data.frame(econ=rep('middle', 15),
                  equip = rep(levels(gavote$equip), rep(3,5)),
                  perAA=rep(c(.11, 0.23, 0.35), 5))
pp <- predict(finalm, new=pdf)

propAA <- gl(3,1,15, labels=c('low', 'medium', 'high'))
xtabs(round(pp, 3) ~ propAA + equip, pdf)
```

```
##         equip
## propAA    LEVER  OS-CC  OS-PC  PAPER  PUNCH
##   low     0.037  0.038  0.045 -0.007  0.031
##   medium  0.032  0.046  0.039  0.003  0.036
##   high    0.027  0.053  0.034  0.014  0.042
```

보면 명확한 상관관계를 파악하기 힘들다.

결론: economic status 는 undercount 에 명확한 상관관계를 보여준다(계층이 낮을수록 높은 관계). Voting Equip 와 African American 의 비율은 모두 undercount 에 영향을 주지만 그 방향은 명확하지 않다.