1. Review of Linear Models: Undercount in '00 US Elections

SM Shafqat Shafiq

2024-12-08

1. Introduction

The 2000 U.S. Presidential election generated much controversy, particularly in the state of Florida where there were some difficulties with the voting machine.

In Meyer (2002), voting data from the state of Georgia is analyzed & presented. The packages used are `faraway` and `dplyr`.

1. Check dataframe:

```r
data("gavote")
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
```

```r
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 0.107 ...
##  $ 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 ...
```

```r
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   : 16926
##  3rd Qu.: 12251
##  Max.   :280975
```

Notes:

1. The indices/row names in the data set are the counties of Georgia, and the variables/columns are things like voting equipment used, percentage of African Americans, et al.

2. We note that some of the variables are factors; i.e. they are categorical variables. Other variables such as PerAA, count, etc, are continuous/integer values.

3. A potential voter goes to the polling station where it is determined whether or not they are registered to vote. If so, a ballot is issued. However, a vote is not counted/recorded if the person:

```
a. Fails to vote for President.
b. Votes for more than one candidate.
c. Voting equipment fails to record the vote.
```

For example, in Appling, 6617 ballots were issued and 6609 votes were cast which means about 518 votes were not recorded. This is called an **Undercount**. The purpose of our analysis will be to determine the factors affecting this **Undercount**.

2. Initial Data Analysis

The first stage in any analysis should be an initial graphical and numerical look at the data. the `summary()` function provides a complete numerical overview.

1. For **Categorical Variables** we get a count of the number of each type that occurs. For example, we notice only 2 counties use paper ballots. This will make it difficult to estimate the effect of this particular voting method on **Undercount**.

2. For **Numerical Variables**, we have 6 summary statistics that are sufficient to get a rough idea of the distributions. In particular, we notice that the number of ballots cast ranges over orders of magnitudes. That is, difference between 1st and 3rd quartiles is huge! And so is the difference between the min/max.

This suggests that we should create a new variable called **Relative Undercount**, rather than use the **(Absolute) Undercount** variable. We do this by **normalizing by the total ballots cast**.

```r
# Absolute Undercount
# Overwriting the original gavote data frame
gavote <- gavote %>%
  mutate(undercount = (ballots - votes) / ballots)

# Now gavote contains the undercount variable.
gavote <- gavote %>% mutate(rel_undercount = sum(ballots - votes)/sum(ballots))
```

```r
str(gavote)
```

```
## 'data.frame':    159 obs. of  12 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 0.107 ...
##  $ 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 ...
##  $ undercount   : num  0.0783 0.0363 0.1052 0.0548 0.0515 ...
##  $ rel_undercount: num  0.0352 0.0352 0.0352 0.0352 0.0352 ...
```

```r
summary(gavote$undercount); summary(gavote$rel_undercount)
```

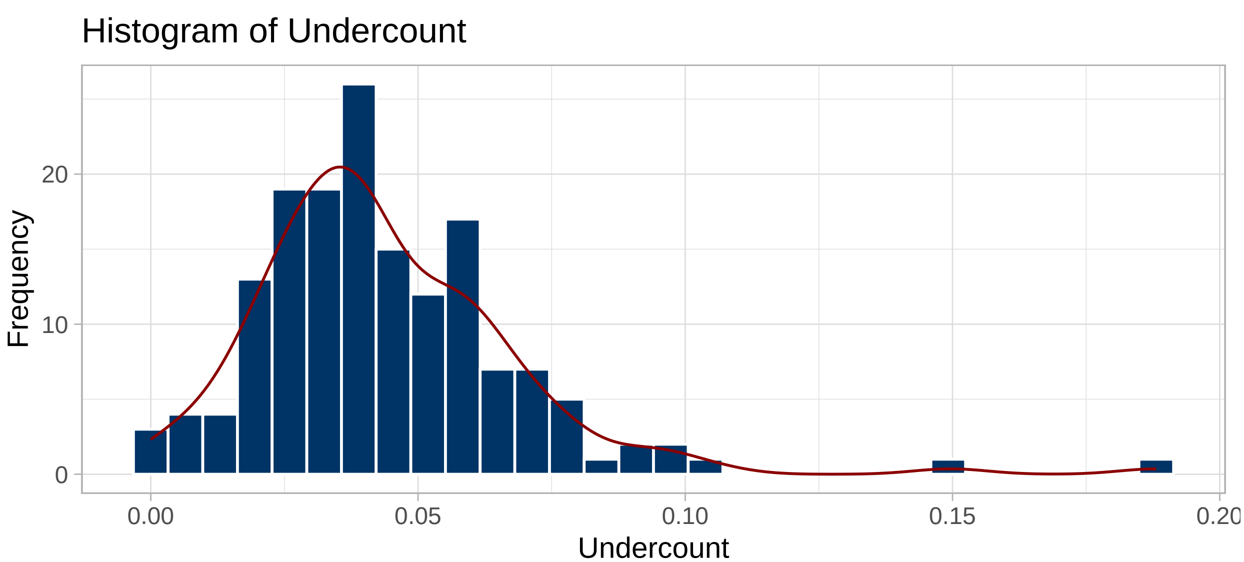
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.02779 0.03983 0.04379 0.05647 0.18812

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.03518 0.03518 0.03518 0.03518 0.03518 0.03518
```

We can see that the **(Relative) Undercount** now ranges from 0 all the way up to almost **19%**. The mean across the counties for **Relative Undercount** is about 4.3% while for **Overall Relative Undercount** it's about 3.5%.

Let's now look at some plots.

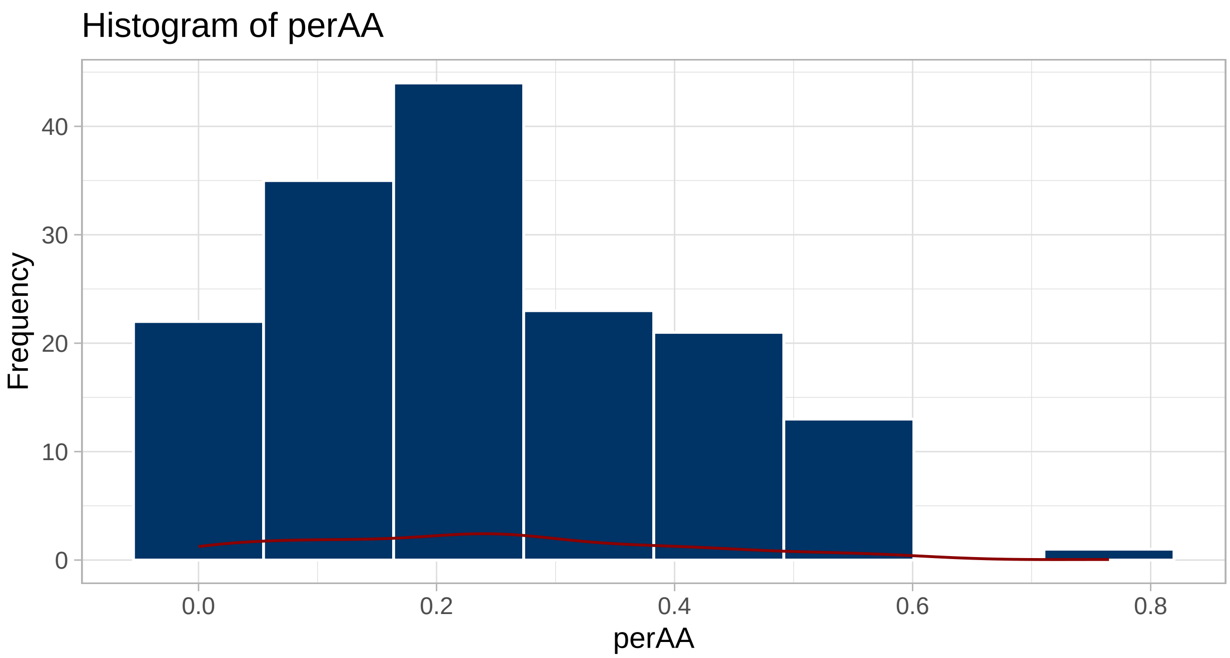
1. Distrubution of Relative Undercount

```r
ggplot(gavote, aes(x = undercount)) +
  geom_histogram(bins = 30, fill = "#003366", color = "white") +
  labs(title = "Histogram of Undercount",
       x = "Undercount", y = "Frequency") +
  geom_density(color = "darkred") +
  theme_light()
```



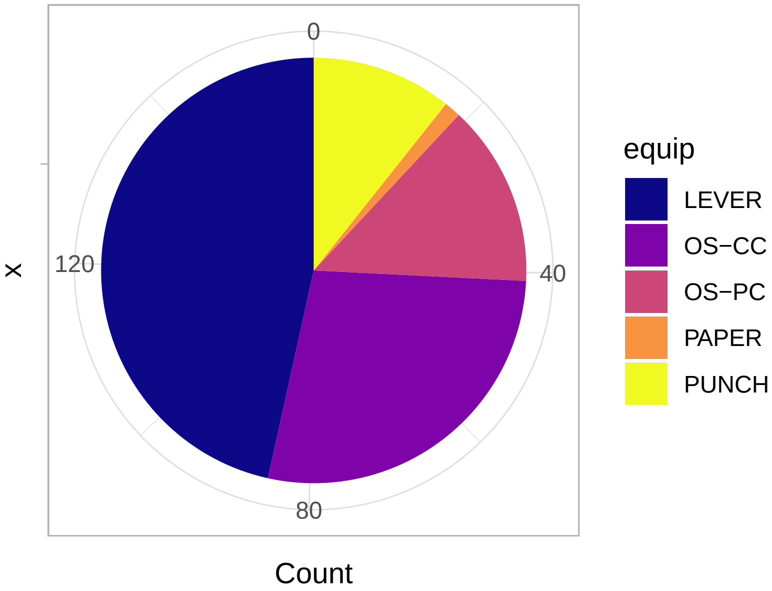
2. Distribution of PerAA

```
ggplot(gavote, aes(x = perAA)) +
  geom_histogram(bins = 8, fill = "#003366", color = "white") +
  labs(title = "Histogram of perAA", x = "perAA", y = "Frequency") +
  geom_density(color = "darkred") +
  theme_light()
```

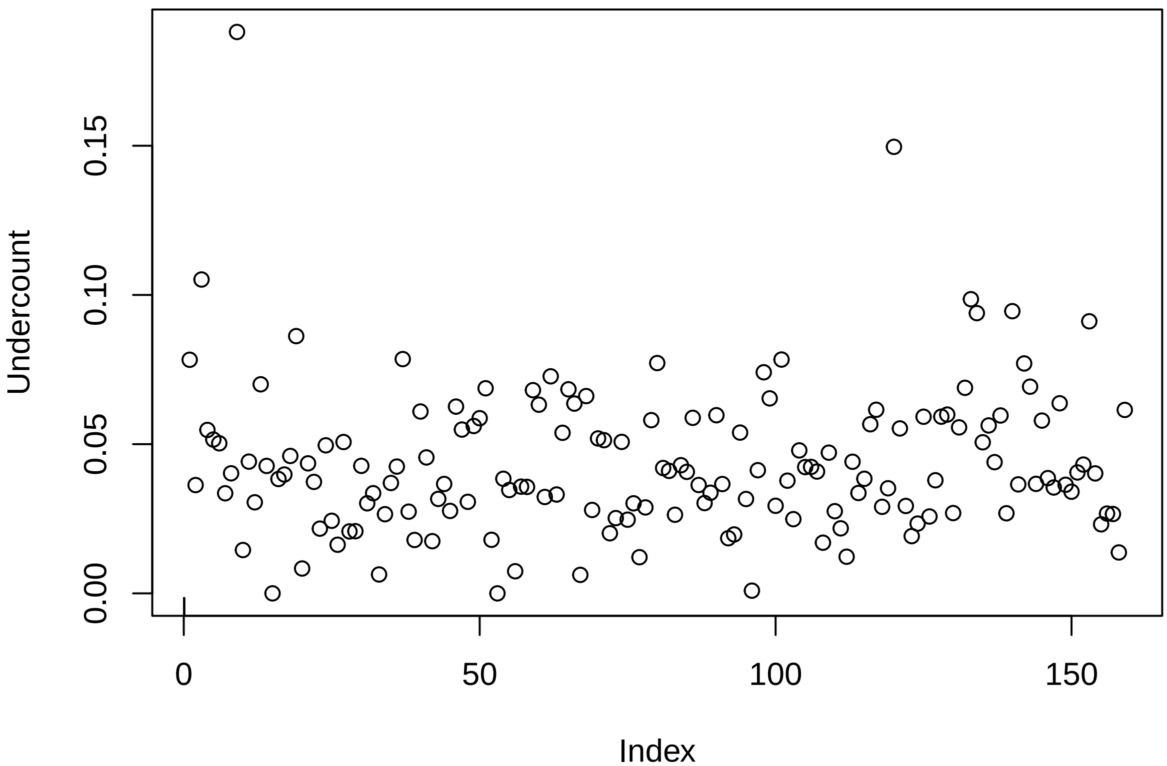


3. Equipment

```
equip <- gavote %>% count(equip)
ggplot(equip, aes(x = "", y = n, fill = equip)) +
  geom_bar(stat = "identity", width = 1) + labs(y = "Count") +
  coord_polar(theta = "y") + theme_light() +
  scale_fill_viridis_d(option = "C")
```



```r
# Rug plot for Undercount
plot(gavote$undercount, ylab = "Undercount")
rug(gavote$undercount)
```



A **histogram** is a crude estimate of the density of the variable and it is sensitive to the choice of bins. A **kernel density** can be as a smoother version of the histogram which is also a superior estimator of density.

A **rug** plot allows us to discern the individual points, and is a good, intuitive way to understand the density and distribution of data points.

What a Rug Plot Tells You:

1. Data Density:
   Closely spaced tick marks indicate a higher concentration of data points in that region. Sparse areas represent lower densities.

2. Outliers:
   Outliers are clearly visible as isolated tick marks far from the main cluster.

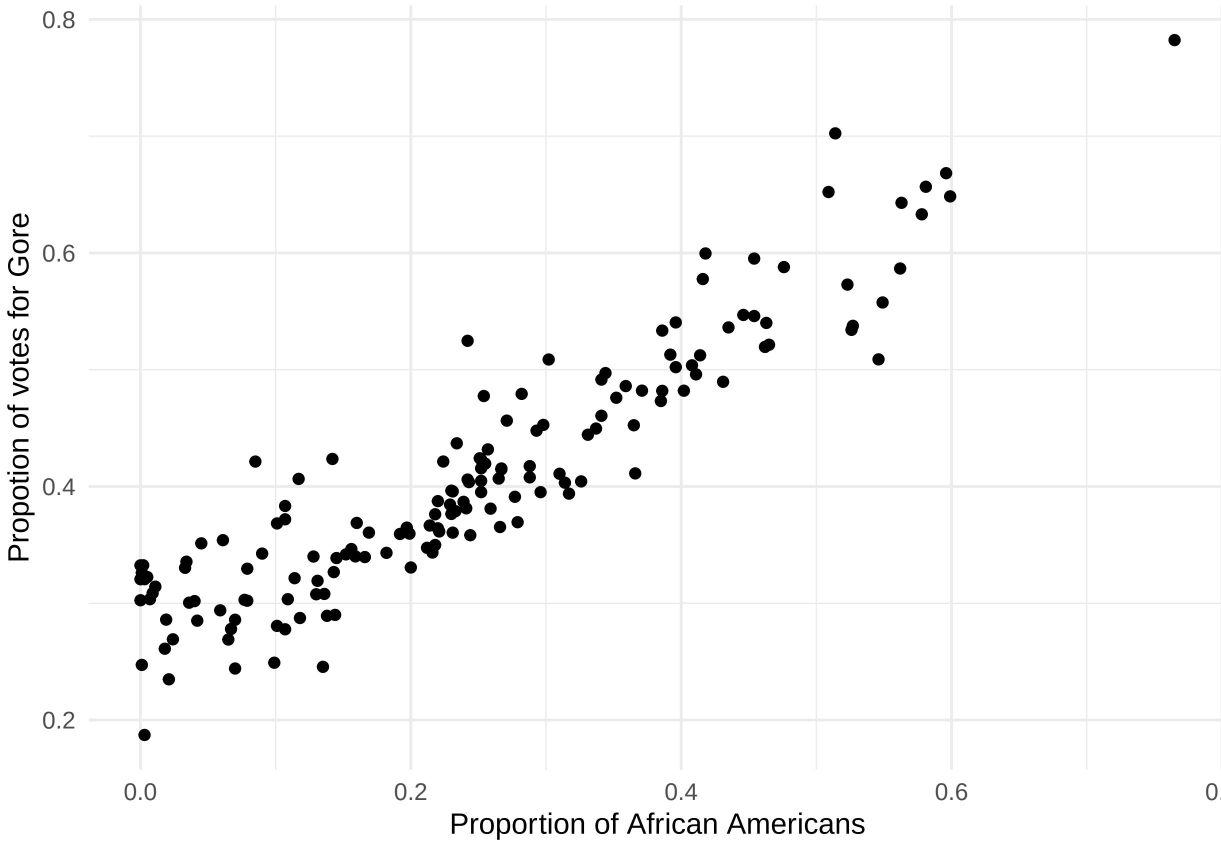
3. Distribution Shape (with Other Plots):
   In a histogram or density plot, the rug plot provides additional detail about individual data points and how well the plot captures the underlying data distribution.

We can see that the distribution is slighlty skewed and there are **2 outliers** in the right tail of the distribution, for Absolute Undercount. These two points are also visible on the rug plot, while the skewness is indicated by the points clustering in the lower half of said rug plot.

We can also view **Categorical Variables** using bar charts, pie charts and so on, while a **scatter plot** is a great way to depict the relationship between **2 numerical variables**.

Let's now look at how **the proportion of voting for Gore relates to the proportion of African Americans**.

```r
gavote$pergore <- gavote$gore/gavote$votes
gavote %>% ggplot(aes(x = perAA, y = pergore)) +
  geom_point() + labs(x = "Proportion of African Americans",
                      y = "Propotion of votes for Gore") +
  theme_minimal()
```

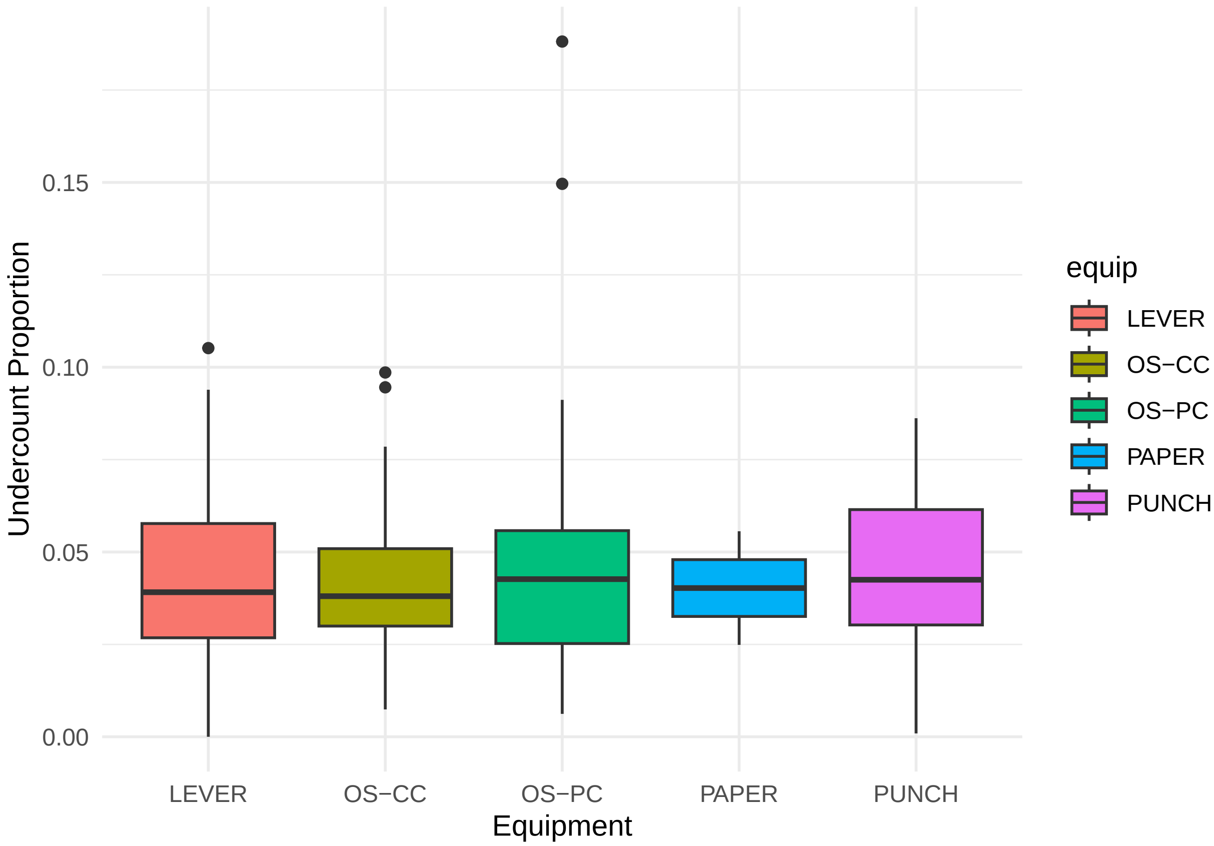


Even though the plot shows a strong positive correlation, in statistical terms it's called an **Ecological Correlation** since **the data points are aggregated across counties**. The plot in itself does not prove that individual African Americans were more likely to vote for Gore, although we know this to be true from other sources.

Side-by-Side plots are another way of displaying the relationship between both **Qualitative/Quantitative** variables,

```r
# Boxplot of Undercount as categorized by equipment used:
gavote %>% ggplot(aes(x = equip, y = undercount, fill = equip)) +
  geom_boxplot() + theme_minimal() +
  labs(x = "Equipment", y = "Undercount Proportion")

# Tabulate 2 categorical variables: Atlanta and Rural
xtabs(~atlanta+rural, gavote)
```

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



We see that the 2 outliers that were detected above were due to the equipment type **OS-PC**. Otherwise, we see no major differences in the proportions of undercount among the equipments. In terms of tabulation, we see that Atlanta had just 1 rural county and 14 urban counties.

We can also check correlation using the `cor()` function for the following variables:

```
gavote %>% select(perAA, ballots, undercount, pergore) %>% cor()
```

```
##                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
```

2. Fitting a Linear Model

We would now like to fit a linear model to determine the factors affect `undercount` in a county. Here, `undercount` is the target variable, and the predictor variables are any of the remaining columns in the dataset. We begin with `pergore` and `perAA` as possible predictor variables, and will fit a linear regression model $Y = \beta \cdot X + \epsilon$ to analyse the possible effects they have on `undercount`.

Our initial model will look like this:

$$\widehat{\text{undercount}} = B_0 + B_1 \cdot \text{pergore} + B_2 \cdot \text{perAA} + \epsilon$$

Here, ϵ is the error/random term.

Gauss-Markov Assumptions:

There are 6 key assumptions of an Ordinary Least Squares model (Gauss-Markov) assumptions. These are as follows:

1. Linearity of the model: $Y = \beta \cdot X + \epsilon$

2. Full Rank Condition: I.e. no multicollinearity. The matrix $X \in \mathbb{R}^{n \times p}$ must have **full column rank**. I.e. $rank(X) = p$

3. Exogeniety: Zero mean of errors.

$E(\epsilon \mid X) = 0$

That is

$E[Y \mid X] = \beta \cdot X$

4. Homoskedasticity: Constant variance of errors. The error term has a constant variance and are uncorrelated to each other.
$Var(\epsilon \mid X) = \sigma^2 \cdot I$

5. No Autocorrelation: Independence of the error terms.
$Cov(\epsilon_i, \epsilon_j) = 0$

6. Errors are normally distributed: Required for inference.
$\epsilon \sim \mathcal{N}(0, \sigma^2 I)$

Residual Analysis:

We begin by analysing the following:
1. Prediction Error: $Y - \hat{Y} = \epsilon = X \cdot \hat{\beta}$
`predict(lmod)`

2. Residuals: `residuals(lmod)`

3. Residual Sum of Errors: $\epsilon^\top \epsilon$
`deviance(lmod)`

4. Degrees of Freedom: `df.residual(lmod)`

5. Variance of Errors: `sqrt(deviance(lmod)/df.resdiual(lmod))`

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

# Add predictions, prediction errors, and residuals to the data frame
gavote <- gavote %>%
  mutate(
    undercount_pred = predict(lmod),            # Predictions
    pred_error = undercount - undercount_pred, # Prediction errors
    residuals_squared = residuals(lmod)^2      # Residuals
  )

gavote %>% select(undercount, undercount_pred, pred_error, residuals_squared) %>% head(3)
```

```
##          undercount undercount_pred   pred_error residuals_squared
## APPLING  0.07828321      0.04133661  0.036946603     0.00136505146
## ATKINSON 0.03629595      0.04329088 -0.006994927     0.00004892901
## BACON    0.10516881      0.03961823  0.065550577     0.00429687809
```

```r
# Deviance
model_deviance <- deviance(lmod)

# Degrees of Freedom
model_df_residual <- df.residual(lmod)

# Variance of Errors (Standard Deviation of Residuals)
error_variance <- sqrt(model_deviance / model_df_residual)

# Output
list(
  Deviance = model_deviance,
  Degrees_of_Freedom = model_df_residual,
  Residual_Error = error_variance
)
```

```
## $Deviance
## [1] 0.09324918
##
## $Degrees_of_Freedom
## [1] 156
##
## $Residual_Error
## [1] 0.02444895
```

```r
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
```

Deviance is a more general measure of the RSS. For linear models, deviance is the RSS. It measures how well the model fits in an absolute sense, but it does not tell us how well it fits in a relative sense. The popular choice is the R^2.

We see that the R^2 is about 5.3% which means that the model does not fit very well. Another way to think about R^2 is as **the squared correlation between predicted values and response values**.

R^2 is not a good selection criteria, since it never decreases when we add variables. Instead we use the **Adjusted** R^2, which is an appropriate alternative. Here, the R_A^2 is 0.04 (4%).

We now add some categorical/interaction variables to the model. Before we do that, we standardize the `pergore` and `perAA` variables by subtracting their means from them.

```r
gavote <- gavote %>%
  mutate(
    cpergore = pergore-mean(pergore),
    cperAA = perAA - mean(perAA),
    usage = rural
  )

lmod2 <- lm(undercount ~ cperAA + cpergore*usage + equip, gavote)
summary(lmod2)
```

```
##
## Call:
## lm(formula = undercount ~ cperAA + cpergore * usage + equip,
##     data = gavote)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.059530 -0.012904 -0.002180  0.009013  0.127496
##
## Coefficients:
##                       Estimate Std. Error t value             Pr(>|t|)
## (Intercept)           0.043297   0.002839  15.253 < 0.0000000000000002 ***
## cperAA                0.028264   0.031092   0.909               0.3648
## cpergore              0.008237   0.051156   0.161               0.8723
## usageurban           -0.018637   0.004648  -4.009            0.0000956 ***
## equipOS-CC            0.006482   0.004680   1.385               0.1681
## equipOS-PC            0.015640   0.005827   2.684               0.0081 **
## equipPAPER           -0.009092   0.016926  -0.537               0.5920
## equipPUNCH            0.014150   0.006783   2.086               0.0387 *
## cpergore:usageurban -0.008799   0.038716  -0.227               0.8205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02335 on 150 degrees of freedom
## Multiple R-squared:  0.1696, Adjusted R-squared:  0.1253
## F-statistic: 3.829 on 8 and 150 DF,  p-value: 0.0004001
```

The terms `usageurban`, `equipOSCC`, `equipPAPER`, and `equipPUNCH` are all dummy variables that take on 1 when the county is urban or is using that voting method respectively. They take on 0 otherwise.

Multicollinearity: Since we already know that `pergore` and `perAA` are correlated with each other, this gives rise to the issue of collinearity which makes the interpretation of regression coefficients difficult. Furthermore, `perAA` is likely to be also correlated with other socioeconomic variables which might also be related to `undercount`.

For an average number of Gore votes (`cpergore` $= 1$), we would predict a 1.86% lower `undercount` in an urban county compared to a rural county (`usageurban` $= 1$).

In a rural county (`usageurban` $= 0$), we predict that if the proportion of of Gore votes increases by 10%, then `undercount` would increase by 0.08%.

In an urban county, a 10% increase in the proportion of Gore votes would lead to a 0.0056 change in `undercount` in that county:

```r
# cpergore = 0.00824
# cpergore*urbanusage = -0.008799
# 10*(`cpergore`-`cpergore`:`usageurban`)
 10*(0.00824 - 0.00880)
```

```
## [1] -0.0056
```

3. Hypothesis Testing

- We want to test the significance of one, some, or all predictors in a model.
- Assuming errors are IID and independent, we can use the **F-Test**.
- This involves comparing two models: the unrestricted model U_R and the restricted model L_R.
- The restricted model L_R includes linear restrictions on the parameters.
- Let $\dim(U_R) = p$ and $\dim(L_R) = q$.
- Assuming L_R is the correct model, the **F-Statistic** is calculated as:

$$F = \frac{\left(\frac{\text{RSS}_{\text{restricted}} - \text{RSS}_{\text{unrestricted}}}{p-q}\right)}{\left(\frac{\text{RSS}_{\text{unrestricted}}}{n-p}\right)}$$

Where:

- $\text{RSS}_{\text{restricted}}$: Residual Sum of Squares for the restricted model.
- $\text{RSS}_{\text{unrestricted}}$: Residual Sum of Squares for the unrestricted model.
- p: Number of parameters in the unrestricted model.
- q: Number of parameters in the restricted model.
- n: Total sample size.

$$F \sim F_{(p-q),(n-p)} \quad \text{under } H_0$$

Where:
- $p - q$: Degrees of freedom for the numerator.
- $n - p$: Degrees of freedom for the denominator.

Therefore, an **F-Test** would involve comparing the calculated **F-Statistic** against the theoretical **F-Score** from the **F-Distribution**.

Decision Rule for the F-Test

- **Reject the null hypothesis** (H_0) if:

$$F_{\text{calculated}} > F_{\text{critical},\alpha,(p-q),(n-p)}$$

- **Fail to reject** H_0 if:

$$F_{\text{calculated}} \leq F_{\text{critical},\alpha,(p-q),(n-p)}$$

Where:

- α: Significance level (e.g., 0.05 for a 5% level).

- $p - q$: Degrees of freedom for the numerator (number of restrictions).

- $n - p$: Degrees of freedom for the denominator (residual degrees of freedom in the unrestricted model).

A larger $F_{\text{calculated}}$ indicates stronger evidence against H_0, suggesting that the restrictions imposed by the restricted model are invalid.

Theorem: F-Statistic Decision Rule

Given:

- Two models: unrestricted model (U_R) and restricted model (L_R).
- Hypothesis:
  - H_0: The restricted model (L_R) is true.
  - H_a: The unrestricted model (U_R) is better (i.e., at least one restriction in L_R is invalid).

If-Else Rule:

1. **If**

$$F = \frac{\left(\frac{\text{RSS}_{\text{restricted}} - \text{RSS}_{\text{unrestricted}}}{p-q}\right)}{\left(\frac{\text{RSS}_{\text{unrestricted}}}{n-p}\right)} > F_{\alpha,(p-q),(n-p)}$$

   Then
   - Reject H_0: The unrestricted model U_R is a significantly better fit.
   - At least one restriction in L_R is invalid.
2. **Else**
   - Fail to reject H_0: The restricted model L_R is adequate.
   - The restrictions are valid, and L_R provides a sufficient explanation.

Where:

- F: Computed F-statistic.
- $F_{\alpha,(p-q),(n-p)}$: Critical value of the F-distribution at significance level α with $(p - q)$ numerator and $(n - p)$ denominator degrees of freedom.
- n: Total sample size.
- p: Number of parameters in U_R.
- q: Number of parameters in L_R.
- $\text{RSS}_{\text{restricted}}$: Residual Sum of Squares for L_R.
- $\text{RSS}_{\text{unrestricted}}$: Residual Sum of Squares for U_R.

Comparing 2 models in R:

** We consider the following 2 models:

- L_R: `lmod1 <- pergore + perAA`
  The restricted model has fewer predictors because it excludes the interaction term usage*equip. A restricted model imposes constraints or restrictions, in this case, by not considering the interaction term.

- U_R: `lmod2 <- pergore + perAA + usage*equip`
  The unrestricted model includes all terms, allowing more flexibility. It is "unrestricted" because it does not impose the restriction of omitting the interaction term.

** We then compare the 2 models (`lmod1`, `lmod2`) using the `anova()` function.

```r
# Restricted Model:
lmod1 <- lm(undercount ~ pergore + perAA, data = gavote)

# Unrestricted Model:
lmod2 <- lm(undercount ~ pergore + perAA + usage*equip, data = gavote)

# Perform ANOVA test to compare the 2 models:
anova(lmod1, lmod2)
```

```
## Analysis of Variance Table
##
## Model 1: undercount ~ pergore + perAA
## Model 2: undercount ~ pergore + perAA + usage * equip
##   Res.Df      RSS Df Sum of Sq     F   Pr(>F)
## 1    156 0.093249
## 2    148 0.079925  8  0.013324 3.084 0.003031 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- The **P-Value** for the unrestricted model U_R here is small (`0.003031 < 0.05`) indicating the **Null Hypothesis** of preferring the smaller, **Restricted Model**, L_R, should be **rejected*.
- The same conclusion can be drawn by looking at the **F-Calculated** (`3.084`), which is greater than the **F-Critical** (`2.43`).

```r
# Degrees of freedom
df1 <- 4  # Numerator degrees of freedom
df2 <- 154  # Denominator degrees of freedom

# Significance level
alpha <- 0.05

# Compute critical F-value
f_critical <- qf(1 - alpha, df1, df2)
f_critical
```

```
## [1] 2.430385
```

- Since **F-Calculated** > **F-Critical**, we reject the null hypothesis of preferring the **Restricted Model** L_R, in favor of the **Unrestricted Model**, U_R.
- It is possible to test specific predictors in a model using the general **F-test Method**:
  - Fit a model with a predictor and w/o it and then compute the **F-Statistic**.
- An alternative is using the **T-test** however we usually try and avoid **T-tests** for **Categorical** variables; instead we use the **F-test**.
- A comparison of all models with one predictor less than the larger may be obtained using the `drop1()` function, setting `test = "F"` in its argument.
  - The `drop1()` function shows the importance of a variable by displaying the impact of dropping it from the model on the AIC.
  - The lower the AIC after dropping the variable, the more important that variable is.
- **Interpretation of AIC Changes:**
  - **Lower AIC After Dropping a Variable:** If dropping a variable leads to a lower AIC, it indicates that the variable is less important (or even detrimental) to the model's fit. In this case, removing it improves the model.

- **Higher AIC After Dropping a Variable:** Conversely, if the AIC increases after dropping a variable, that variable is considered important to the model. Its presence contributes to a better fit, reflected by a lower AIC.

```
drop1(lmod2, test = "F")
```

```
## Single term deletions
##
## Model:
## undercount ~ pergore + perAA + usage * equip
##             Df  Sum of Sq      RSS     AIC F value Pr(>F)
## <none>                    0.079925 -1185.7
## pergore      1 0.00000198 0.079927 -1187.7  0.0037 0.9518
## perAA        1 0.00060256 0.080528 -1186.5  1.1158 0.2925
## usage:equip  3 0.00187824 0.081804 -1188.0  1.1593 0.3274
```

Confidence Intervals

```
confint(lmod2)
```

```
##                               2.5 %      97.5 %
## (Intercept)             0.008996452 0.058276185
## pergore                -0.087773385 0.093324710
## perAA                  -0.028242850 0.093110274
## usageurban             -0.029901198 0.002054072
## equipOS-CC             -0.004258315 0.016881084
## equipOS-PC              0.008790761 0.036240082
## equipPAPER             -0.041684041 0.024716269
## equipPUNCH             -0.008504873 0.030848131
## usageurban:equipOS-CC -0.023228692 0.021252713
## usageurban:equipOS-PC -0.047645052 0.004201165
## usageurban:equipPAPER            NA          NA
## usageurban:equipPUNCH -0.027447700 0.028702528
```

- Confidence intervals have a duality with the corresponding **t-tests** in that if the *p-value* is greater than 5% then 0 will fall in the interval.
- Confidence intervals give a range of plausible values for the parameter and are more useful for judging the size of the effect of the predictor, than the *p-value*.
- These intervals are individually correct, but there is no *95% chance that the true parameter falls in this interval.*

4. Diagnostics

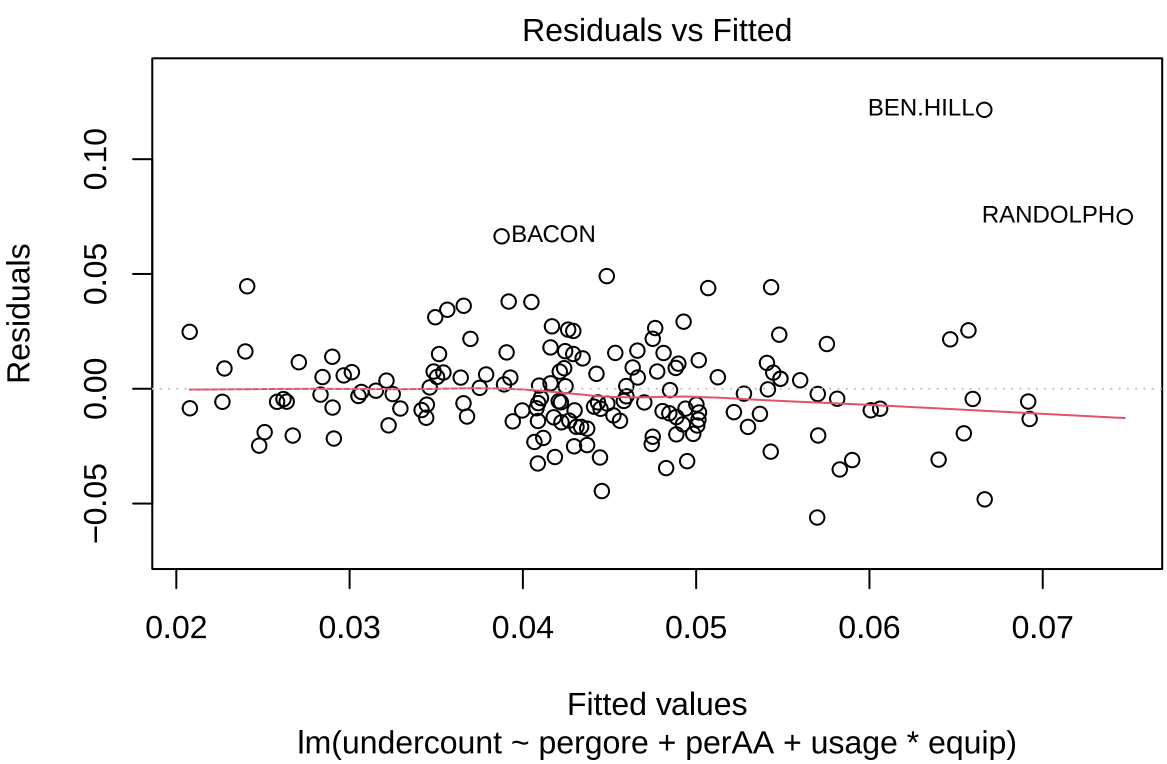
- Validity of inference depends on the assumptions around the linear model.

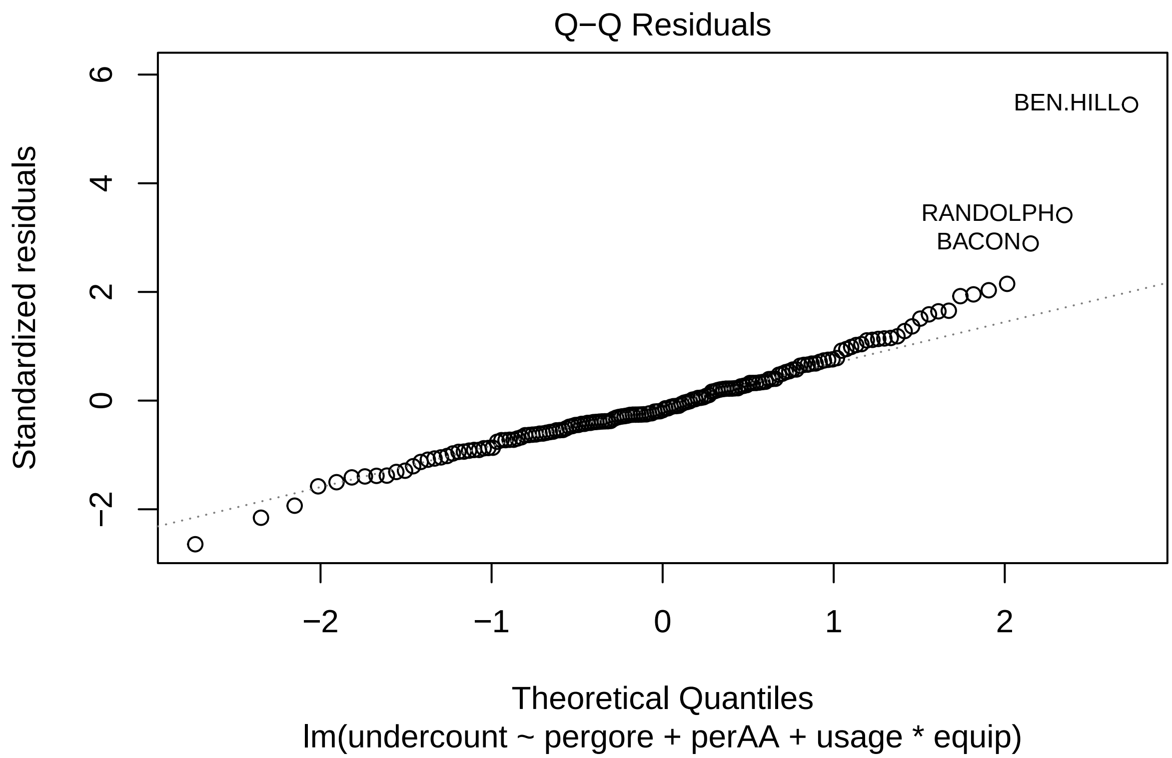
- We are also interested in *outlier*; ideally we'd prefer ** each point to have equal contribution** to the model yet sometimes we find points that have a lerger effect than the other. These are known as **Influential Points**.

- A successful data analyst should pay more attention to avoiding big mistakes than optimizing the fit!
- The `plot(lmod2)` function gives us the following 4 plots:

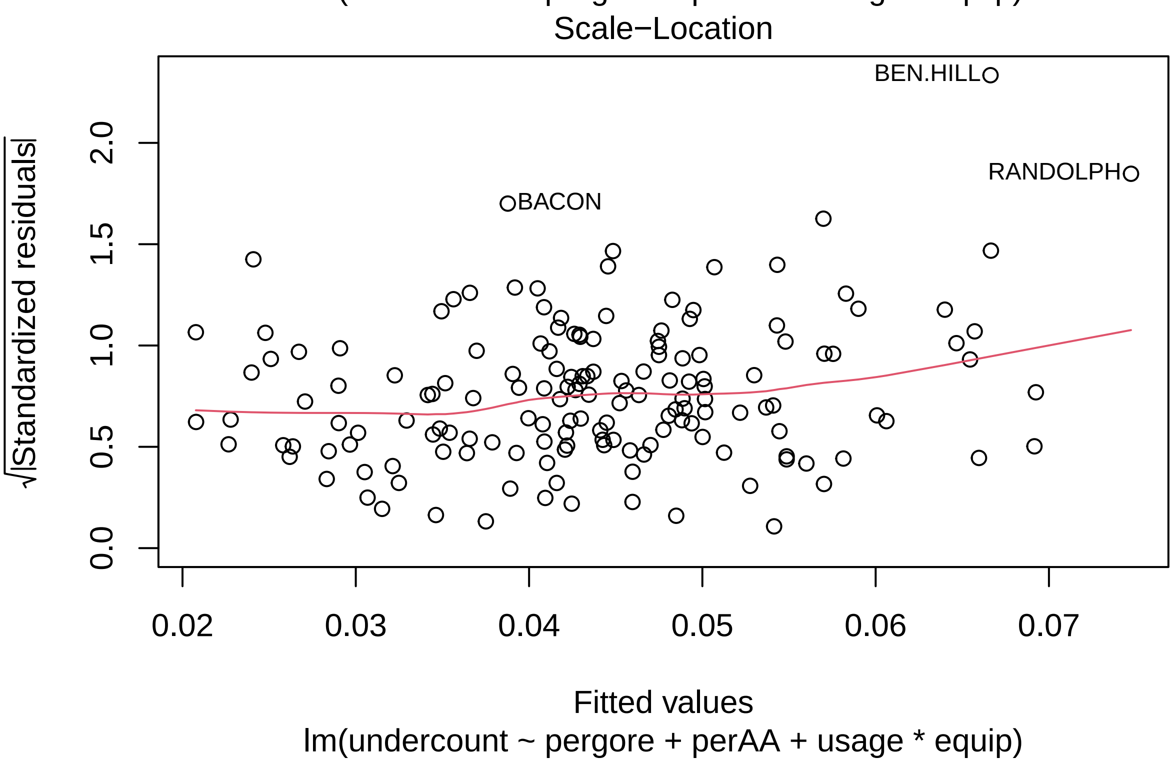
1. **Residual vs Fitted:**
   Detects lack of fit. If the plot shows some curved-linear trend, this is a sign that some change to the model is needed. We can also check covariance assumption of the errors using this.
2. **Normal Q-Q:**
   Assesses the normality of residuals. If the points on the plot lie approximately along the 45-degree reference line, the residuals are normally distributed. Deviations from this line suggest departures from normality, which can affect hypothesis tests and confidence intervals.
3. **Scale - Location:**
   Evaluates the homoscedasticity of residuals. This plot shows the square root of standardized residuals versus fitted values. A horizontal line with equally spread points indicates constant variance. A funnel shape or curvature suggests heteroscedasticity, meaning the variance of residuals changes with the level of fitted values.
4. **Residuals vs Leverage:**
   Identifies influential observations that have a significant impact on the regression model. The leverage measures how far an observation is from the mean of the predictor values. Points with high leverage and large residuals are considered influential. The Cook's distance is often plotted to help assess influence, where values greater than 1 suggest that the observation has a substantial impact on the model.

```r
plot(lmod2)
```

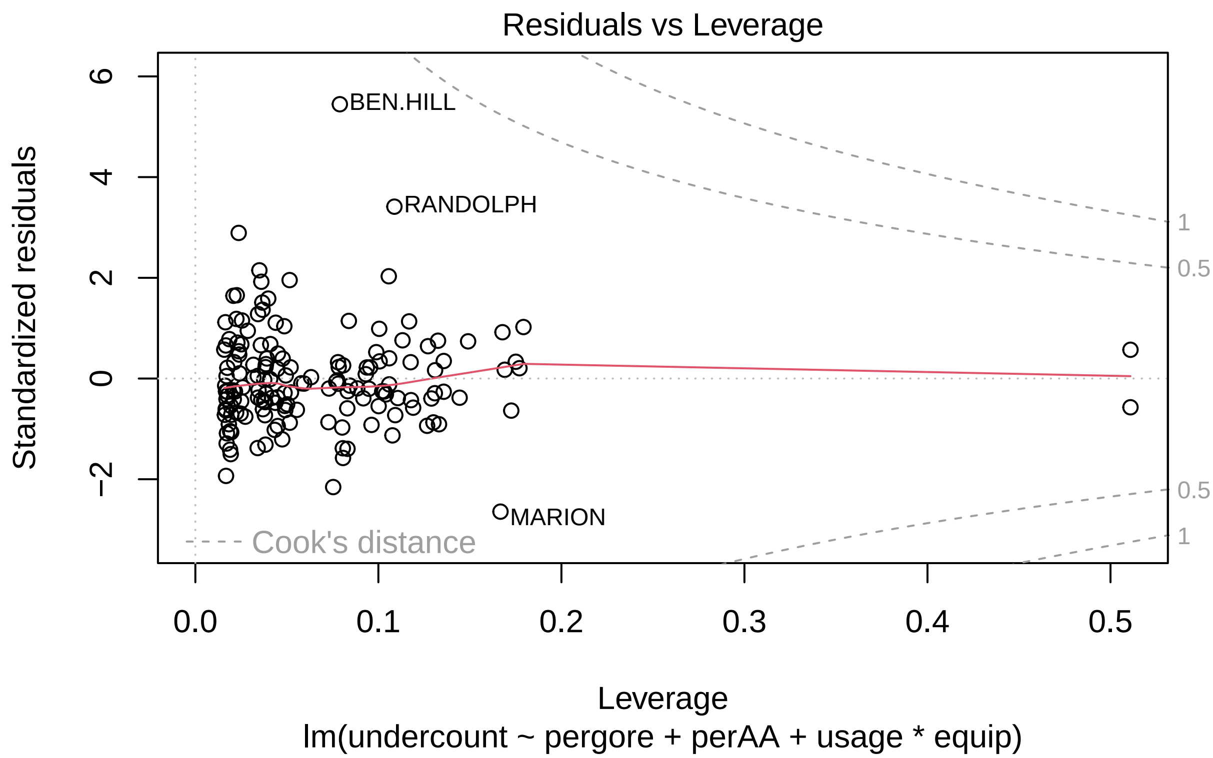




Q−Q Residuals
Standardized residuals
6
4
2
0
−2
BEN.HILL
RANDOLPH
BACON
−2
−1
0
1
2
Theoretical Quantiles
lm(undercount ~ pergore + perAA + usage * equip)



Scale−Location
√|Standardized residuals|
2.0
1.5
1.0
0.5
0.0
BEN.HILL
RANDOLPH
BACON
0.02
0.03
0.04
0.05
0.06
0.07
Fitted values
lm(undercount ~ pergore + perAA + usage * equip)



Hat Matrix (H)

- The **Hat Matrix** is denoted as H and is used in linear regression to project the observed values onto the fitted values.
- It is defined as:

$$H = X(X'X)^{-1}X'$$

  where X is the design matrix of predictors, and X' is its transpose.
- The Hat Matrix has several important properties:
  - **Square Matrix:** H is a square matrix with dimensions equal to the number of observations.
  - **Idempotent:** H satisfies the condition $H^2 = H$, meaning applying it twice has the same effect as applying it once.
  - **Symmetric:** H is symmetric, meaning $H' = H$.

Leverage

- **Leverage** measures the influence of each observation on the fitted values of the model. It quantifies how much an observation affects the estimation of the regression coefficients.
- Leverage values are derived from the Hat Matrix:

$$h_i = H_{ii}$$

  where h_i is the leverage of the i-th observation.
- The leverage values range from 0 to 1:
  - **Low Leverage (close to 0):** The observation has little influence on the fitted values.
  - **High Leverage (close to 1):** The observation is far from the mean of the predictor variables, potentially having a significant impact on the model.

Identifying Influential Observations

- Observations with high leverage (typically $h_i > \frac{2p}{n}$, where p is the number of predictors and n is the number of observations) should be examined closely.
- Influential points can distort regression results, affecting the estimated coefficients and overall fit of the model.

Visualization

- Leverage can be visualized using the **Residuals vs Leverage** plot, which helps to identify influential observations.
- The plot typically includes:
  - **Cook's Distance** lines, indicating influential points (where Cook's Distance > 1).
  - A reference line for high leverage thresholds to assess potential influential data points.

Cook's Distance greater than `0.1` or **Hat Values** greater than `0.3` can be used to identify influential points with high leverage(s), and warrant further investigation.

```r
# Identify Influential points using Cook's Distance > 0.1
gavote[cooks.distance(lmod2) > .1, c(1, 2, 3, 4, 13, 11)]
```

```
##          equip econ perAA rural   pergore   undercount
## BEN.HILL OS-PC poor 0.282 rural 0.4792963 0.1881205365
## MARION   PUNCH poor 0.337 rural 0.4496337 0.0009149131
## RANDOLPH OS-PC poor 0.527 rural 0.5375633 0.1496193313
```

```r
# Identify Influential points using Hat Values > 0.3
gavote[hatvalues(lmod2) > .3, c(1, 2, 3, 4, 13, 11)]
```

```
##            equip econ perAA rural   pergore undercount
## MONTGOMERY PAPER poor 0.243 rural 0.4037465 0.02487369
## TALIAFERRO PAPER poor 0.596 rural 0.6682692 0.05561862
```

We can see that by using both the diagnostic plots and the `cooks.distance()` and `hatvalues()` functions, we can identify the most influential points or those with high leverage. The following observations are made:

- Interestingly, all five counties—**Montgomery**, **Taliaferro**, **Ben Hill**, **Marion**, and **Randolph**—are classified as **poor**.

- **Montgomery** and **Taliaferro** used **PAPER** as their voting equipment, while **Ben Hill** and **Randolph** used **OS-PC**.

- **Marion** utilized **PUNCH** as their voting method.

- **Ben Hill** has the highest percentage of **undercount** at **18.81%**.

These factors could contribute to the observations being classified as either influential points or points with high leverage.

5. Partial Regression Plots

- *Partial Regression Plot* shows the marginal relationship between X_i and Y_i.

- If the relationship is linear, then no transformations are necessary.

- If the **Standardized Residual** is greater than `3.5`, then investigate.

Steps to plot and analyse Partial Regression Plots

1. **Install and Load the Required Package:**

    - Ensure the `visreg` package is installed and loaded.

```
library(visreg)
```

2. **Fit a Linear Model:**
    - Use the `lm()` function to fit a linear regression model with your response variable and predictors.

    ```
    lmod2 <- lm(undercount ~ pergore + perAA + usage:equip, data = gavote)
    ```

3. **Visualize Partial Regression Plots:**
    - Use the `visreg()` function to create partial regression plots for the desired predictor.

    ```
    visreg(model, "predictor_name", gg = TRUE)
    ```

```
visreg(lmod2, "pergore", gg = TRUE)
```

```
## Conditions used in construction of plot
## perAA: 0.233
## usage: rural
## equip: LEVER
```

```
visreg(lmod2, "perAA", gg = TRUE)
```

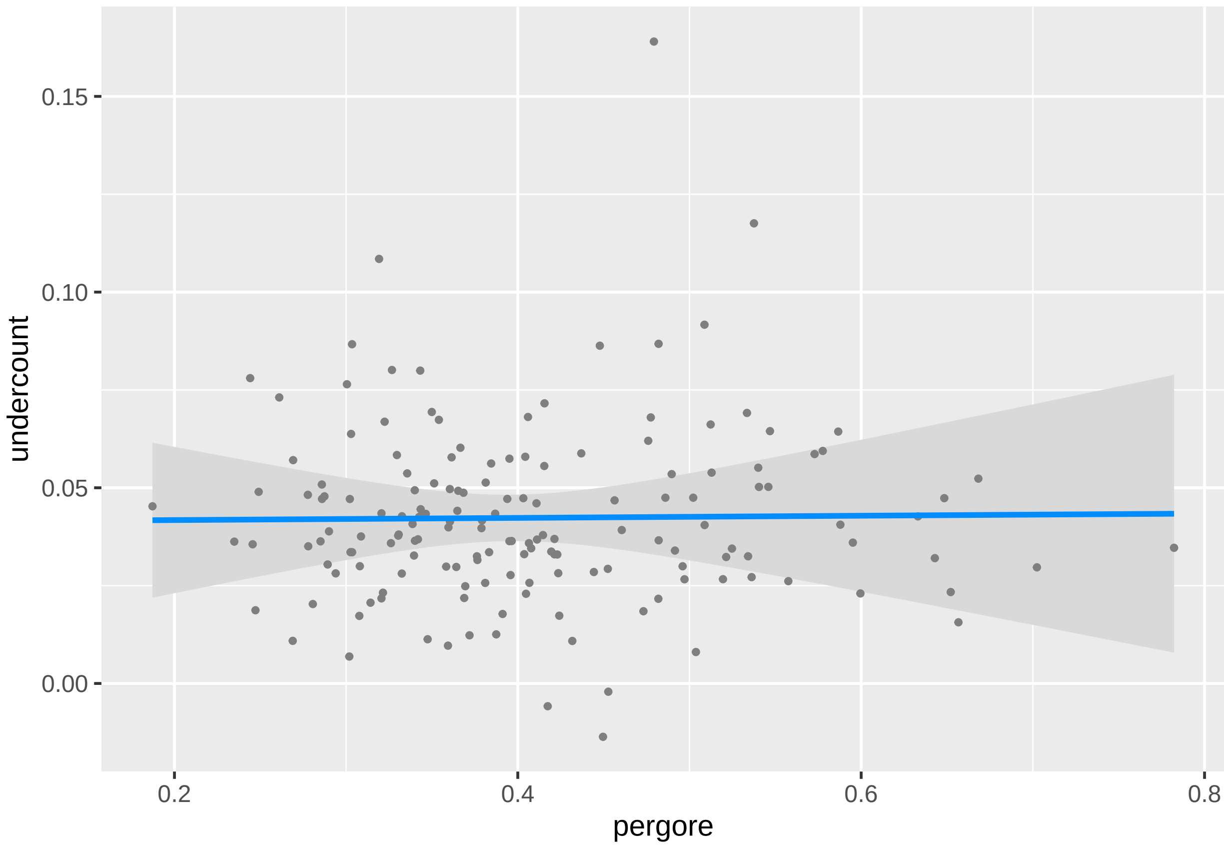
```
## Conditions used in construction of plot
## pergore: 0.3874474
## usage: rural
## equip: LEVER
```

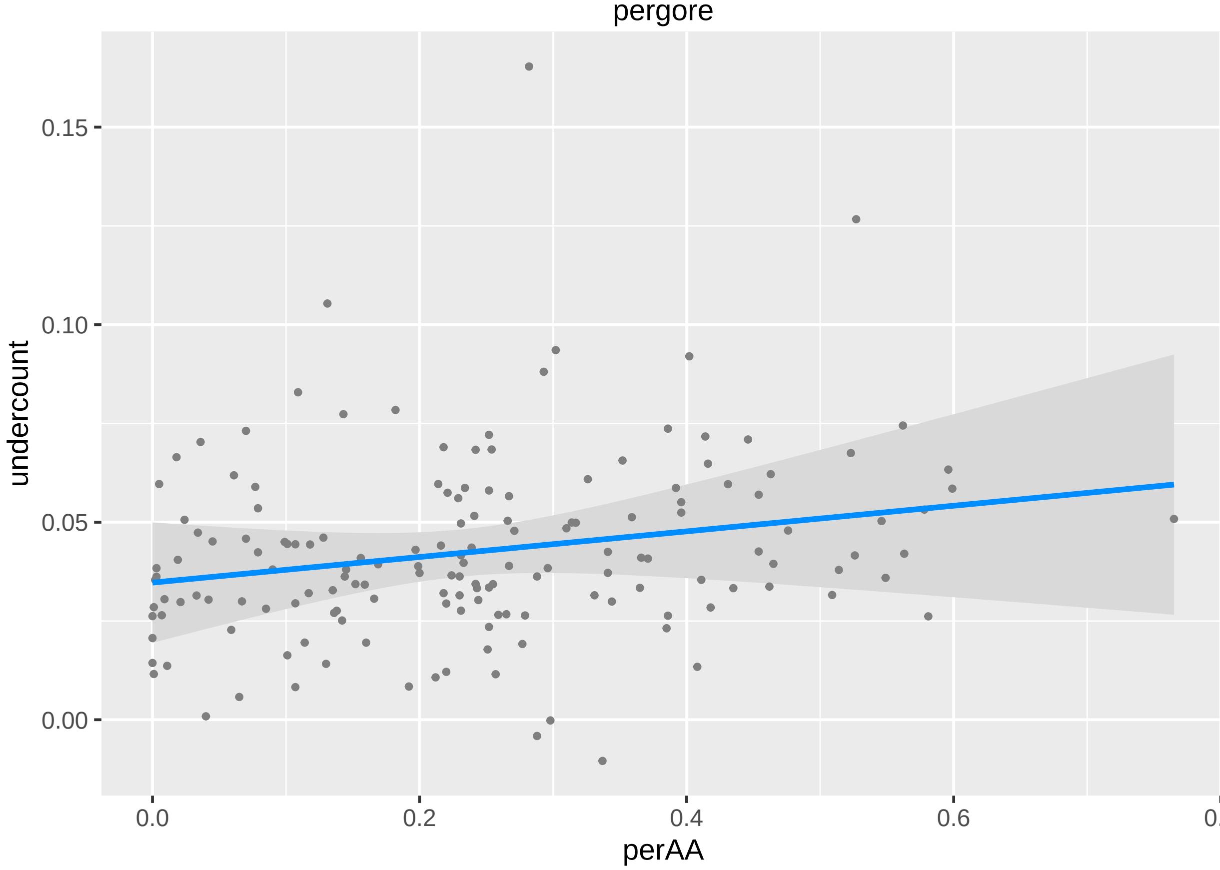
```
visreg(lmod2, "usage", gg = TRUE)
```

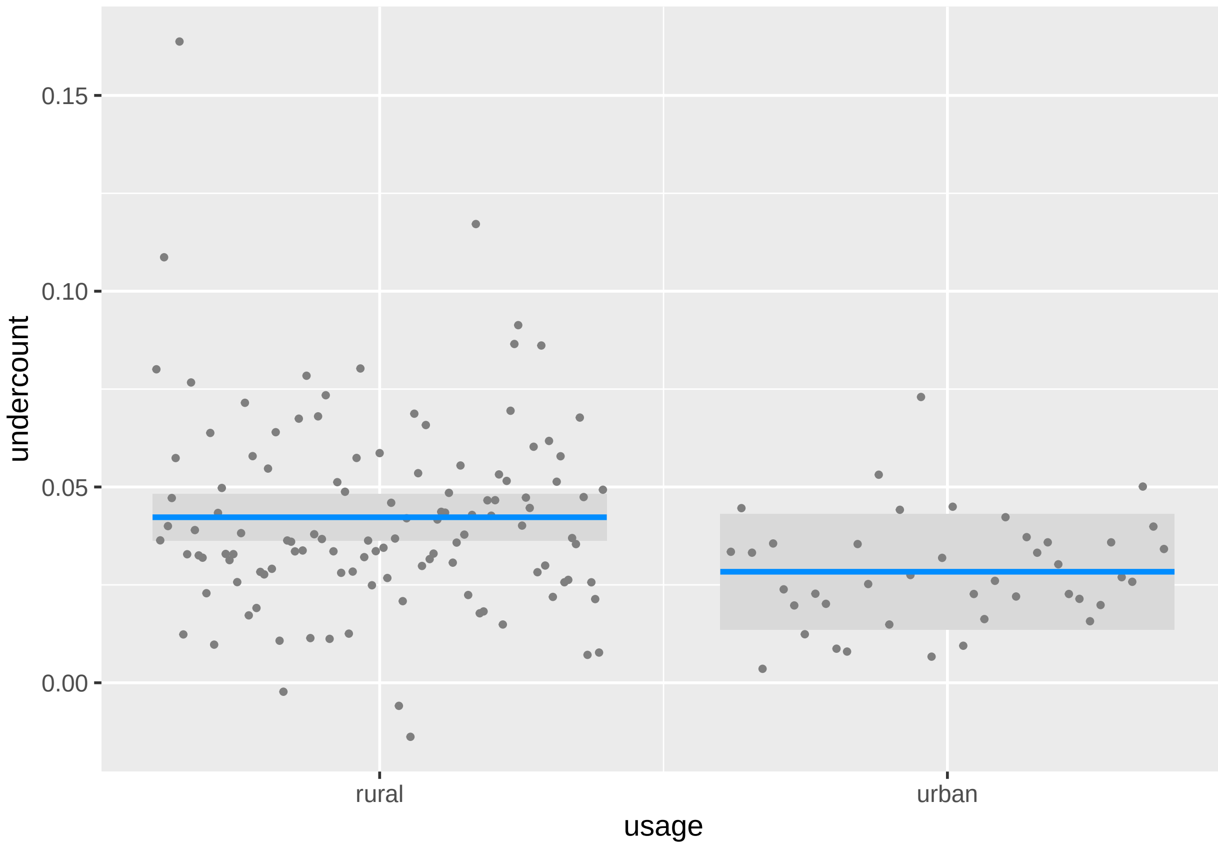
```
## Conditions used in construction of plot
## pergore: 0.3874474
## perAA: 0.233
## equip: LEVER
```

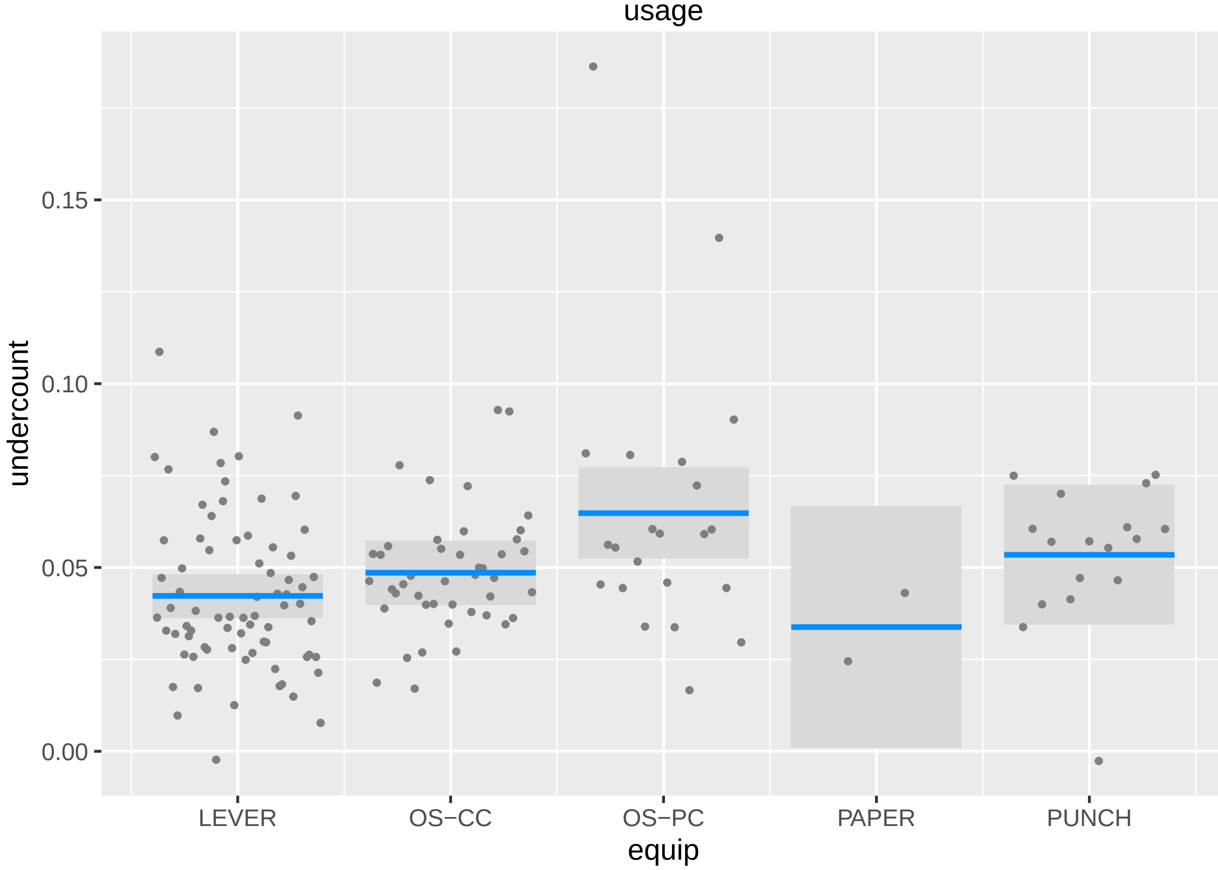
```
visreg(lmod2, "equip", gg = TRUE)
```

```
## Conditions used in construction of plot
## pergore: 0.3874474
## perAA: 0.233
## usage: rural
```


0.15
0.10
0.05
0.00
undercount
0.2
0.4
0.6
0.8
pergore


0.15
0.10
0.05
0.00
undercount
0.0
0.2
0.4
0.6
0.8
perAA


0.15
0.10
0.05
0.00
undercount
rural
urban
usage


0.15
0.10
0.05
0.00
undercount
LEVER
OS−CC
OS−PC
PAPER
PUNCH
equip

4. **Customize the Plot (Optional):**
    - Add optional arguments to customize the visualization, such as `rug = TRUE` for rug plots or `line = list(col = "color")` to change line color.

5. **Interpret the Results:**

- **Straight Line for `undercount ~ pergore`:**
  - A straight line indicates a constant relationship between `undercount` and `pergore` after accounting for the effects of other predictors in your model. This suggests that `pergore` does not contribute significantly to explaining the variability in `undercount`. Essentially, changes in `pergore` do not correspond to changes in `undercount`, indicating that `pergore` may not be an important predictor in the context of your model.
- **Increasing Line for `undercount ~ perAA`:**
  - An increasing line indicates a positive relationship between `undercount` and `perAA`. This means that as `perAA` increases, `undercount` also tends to increase, suggesting that `perAA` is positively associated with `undercount`. This implies that `perAA` is likely an important predictor, as it appears to influence the response variable in a meaningful way.
- **`usage`:**
  - Higher proportion of `undercount` for rural counties.

  - Lower proportion of `undercount` for urban counties.
- **`equip`:**
  - Same interpretation as the boxplots

Summary

- **Constant Relationship**: The straight line for `pergore` suggests it may not be a significant predictor of `undercount`.
- **Positive Relationship**: The increasing line for `perAA` suggests a significant positive association with `undercount`, indicating that changes in `perAA` are related to changes in `undercount`.

These interpretations can help you understand the role of these predictors in your model and guide further analysis or model adjustments.

Standardized Residuals

Standardized residuals are useful for identifying outliers and assessing the influence of individual observations in a regression model. They are calculated using the following formula:

$$r_i = \frac{e_i}{\hat{\sigma}\sqrt{1 - h_i}}$$

Where:

- r_i = standardized residual for observation i
- e_i = raw residual for observation i (the difference between the observed value and the predicted value)
- $\hat{\sigma}$ = standard deviation of the residuals
- h_i = leverage of observation i (the diagonal element of the hat matrix)

Steps to Compute Standardized Residuals in R

1. **Fit a linear model** using the `lm()` function.
2. **Obtain the residuals** and leverage values.
3. **Calculate the standard deviation of the residuals**.
4. **Compute standardized residuals** using the formula above.

```r
# Get the residuals
residuals <- resid(lmod2)

# standard_deviation <- sd(residuals)
standard_deviation <- sd(residuals)

# standardized_residuals_manual <- residuals / standard_deviation
gavote$standardized_residuals <- residuals/standard_deviation

# Identify those points with Standardized Residuals > 3.5
gavote[gavote$standardized_residuals > 3.5, c(1, 2, 3, 4, 13, 11)]
```

```
##          equip econ perAA rural   pergore undercount
## BEN.HILL OS-PC poor 0.282 rural 0.4792963  0.1881205
```

6. Robust and Weighted Least Squares

1. **Robust Regression:**

Least squares works well when there are normal errors, but performs poorly for *long-tailed errors*.

When you have identified a few potential outliers in the current model, one approach would be to simply eliminate them from the dataset and proceed with least squares. This approach only works when we are convinced that the outliers are incorrect observations.

Another approach which is preferred, is the *Robust Least Squares* method that downweights the effects of large errors.

```r
library(MASS)

rlmod2 <- rlm(undercount ~ perAA + pergore*usage + equip, data = gavote)

summary(rlmod2)
```

```
##
## Call: rlm(formula = undercount ~ perAA + pergore * usage + equip, data = gavote)
## Residuals:
##          Min           1Q       Median           3Q          Max
## -0.060257913 -0.011648649 -0.000006587  0.010997677  0.137943406
##
## Coefficients:
##                     Value   Std. Error t value
## (Intercept)          0.0368  0.0121     3.0381
## perAA                0.0327  0.0254     1.2897
## pergore             -0.0082  0.0418    -0.1972
## usageurban          -0.0197  0.0128    -1.5420
## 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
## pergore:usageurban  0.0073  0.0316     0.2298
##
## Residual standard error: 0.01722 on 150 degrees of freedom
```

2. **Weighted Least Squares:**

The sizes of the counties vary greatly, with the number of ballots cast in each county ranging from `881` to `280,975`.

We can determine how much a variable varies by either looking at the standard deviation or more explicitly, computing its *Coefficient of Variation* which is given as follows:

$$CV = \frac{\text{Standard Deviation}}{\text{Mean}} \times 100$$

If the `CV` is equal to or greater than `1`, it means the data has a lot of variability/spread.

```r
# Check the standard deviation of ballots
gavote %>% summarise(ballots_sd = sd(ballots, na.rm = TRUE))
```

```
##   ballots_sd
## 1   37865.15
```

```r
# Calculate the coefficient of Variation
sd(gavote$ballots, na.rm = TRUE) / mean(gavote$ballots, na.rm = TRUE)
```

```
## [1] 2.237033
```

We might expect the *proportion of undercounted votes* to be more variable in smaller counties & since response from larger counties might be more precise (due to CLT), perhaps they should contribute more. This can be verified by creating a new categorical variable called `county_type` which is either `large` or `small` based on the total number of votes, and then performing a grouped `sd()` of `undercount` based on `county_type`.

We can configure the regression to reflect more contribution from larger counties by using *Weighted Least Squares*, where we attempt to minimize:

$$\sum_{i=1}^{n} w_i \epsilon_i^2$$

where $\epsilon_i = y_i - \hat{y}_i$ is the residual for each observation, and the weights are defined as:

$$w_i = \frac{1}{\text{Var}(y_i)}$$

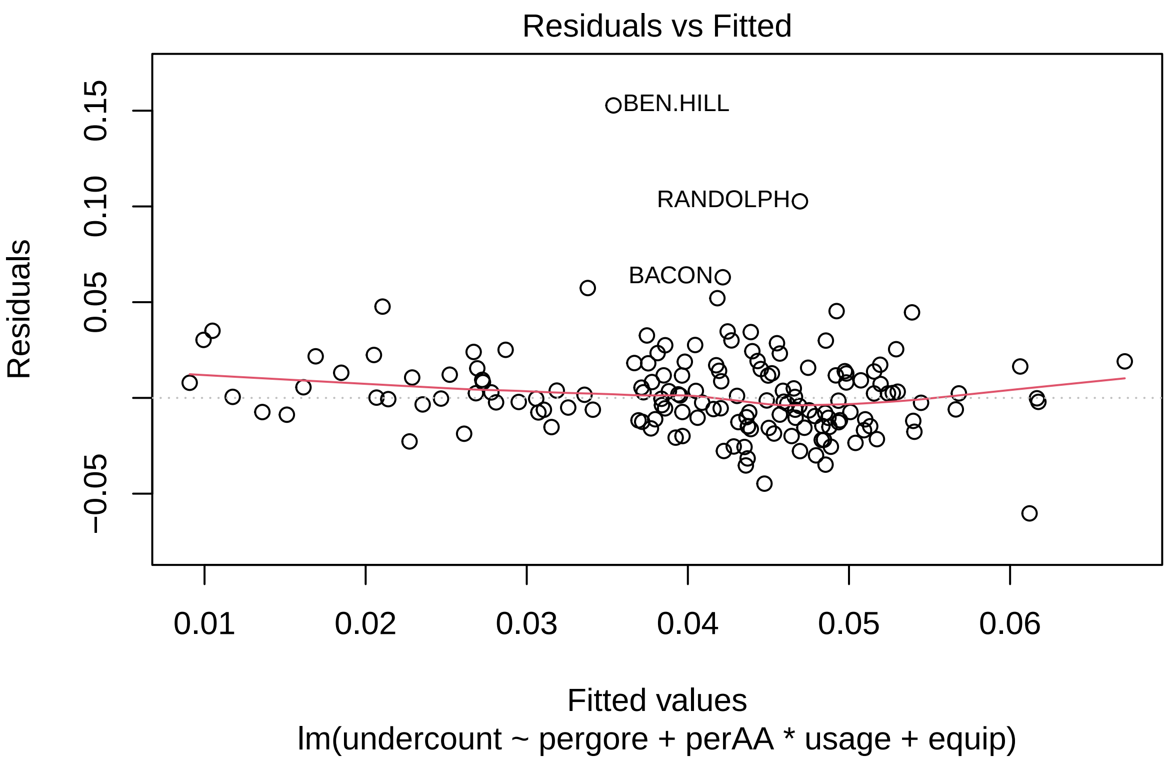
$Var(y_i)$ for binomial response variable is inversely proportional to the group size which here is the number of ballots. This suggest setting the weights proportional to the number of ballots.

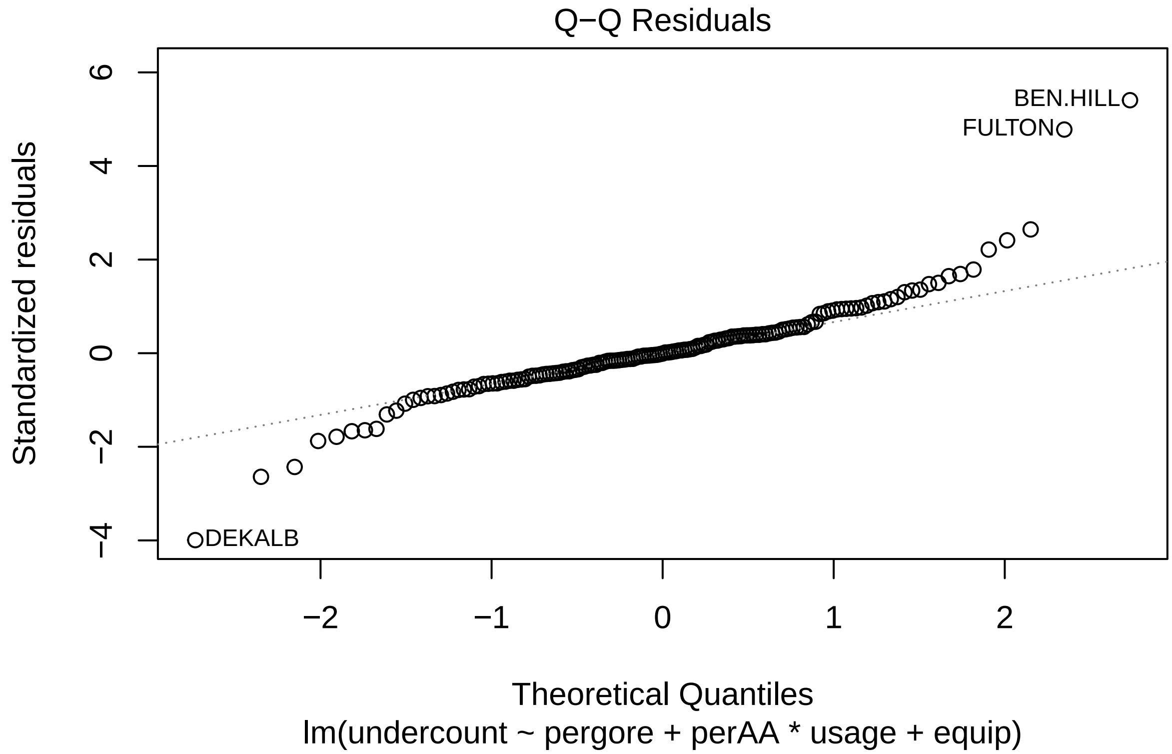
```r
wlmod2 <- lm(undercount ~ pergore + perAA*usage + equip, gavote, weights = ballots)

summary(wlmod2)
```

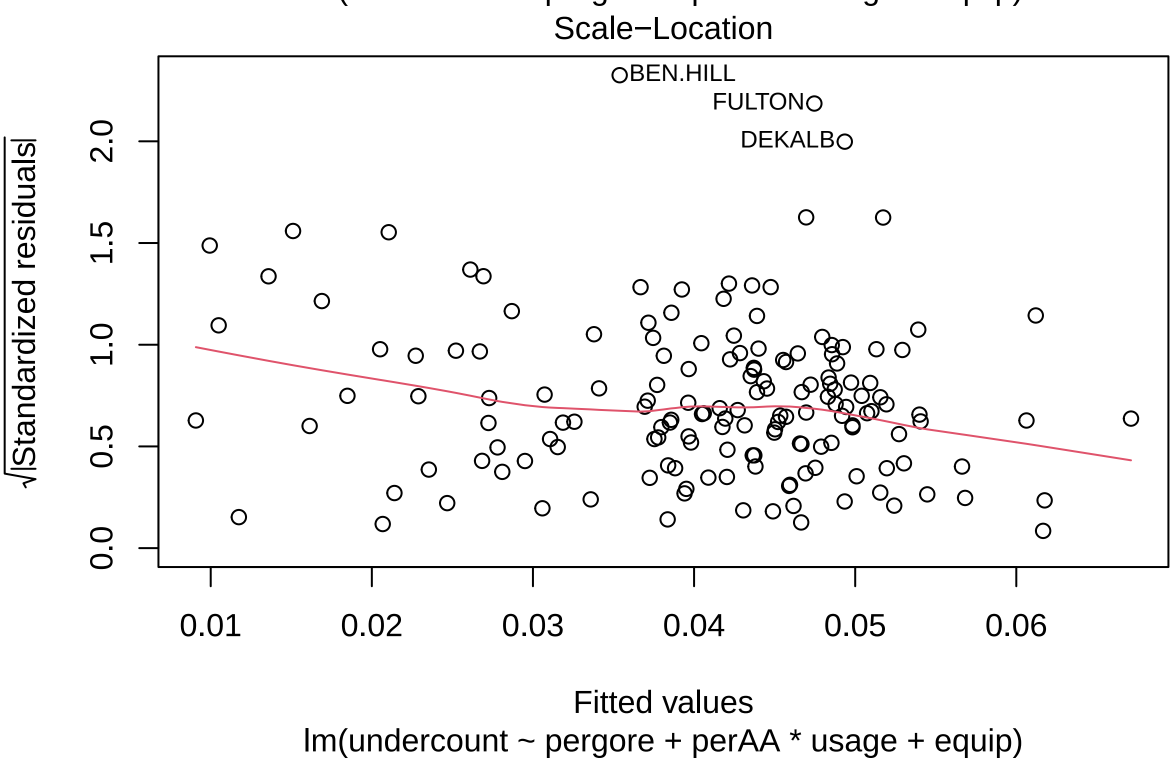
```
##
## Call:
## lm(formula = undercount ~ pergore + perAA * usage + equip, data = gavote,
##     weights = ballots)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0643 -0.9473 -0.0153  0.9637 11.5729
##
## Coefficients:
##                      Estimate Std. Error t value  Pr(>|t|)
```

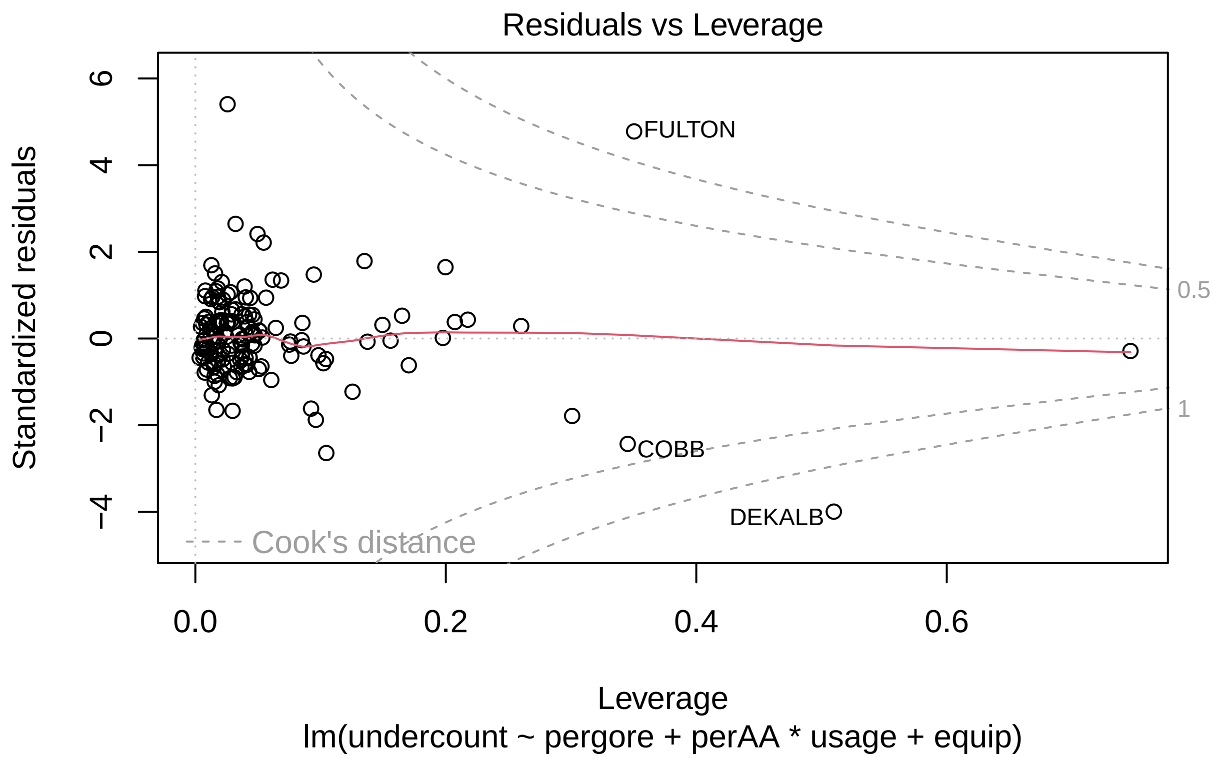
```
## (Intercept)        0.052250   0.011926   4.381 0.0000221 ***
## pergore           -0.056496   0.036709  -1.539    0.1259
## perAA              0.060675   0.025603   2.370    0.0191 *
## usageurban        -0.023995   0.005730  -4.187 0.0000479 ***
## equipOS-CC         0.004503   0.004695   0.959    0.3391
## equipOS-PC        -0.006900   0.004706  -1.466    0.1447
## equipPAPER        -0.013120   0.037093  -0.354    0.7241
## equipPUNCH         0.013912   0.005423   2.565    0.0113 *
## perAA:usageurban  0.030507   0.024320   1.254    0.2116
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.169 on 150 degrees of freedom
## Multiple R-squared:  0.4156, Adjusted R-squared:  0.3844
## F-statistic: 13.33 on 8 and 150 DF,  p-value: 0.00000000000001847
```

```r
plot(wlmod2)
```




Q−Q Residuals
Standardized residuals
BEN.HILL
FULTON
DEKALB
Theoretical Quantiles
lm(undercount ~ pergore + perAA * usage + equip)


Scale−Location
√|Standardized residuals|
BEN.HILL
FULTON
DEKALB
Fitted values
lm(undercount ~ pergore + perAA * usage + equip)



The new model is dominated by data from a few large counties. Compare this against the standard residuals and if they're significantly different then we leave the WLS.

7. Variable Selection

We use the `Akaike Information Criterion` for model/variable selection purposes. The AIC is given as follows:

$$AIC = 2P - 2log(MLE)$$

- The goal is to find the model with the smallest/lowest AIC which also will have the largest MLE. We use the `step()` function to achieve this goal.

```r
library(broom)
biglm <- lm(undercount ~ (equip + econ + usage + atlanta)^2 +
              (equip + usage + atlanta)*(perAA + pergore), gavote)

smallm <- step(biglm, trace = FALSE)

tidy_results <- tidy(smallm)
print(tidy_results)
```

```
## # A tibble: 22 x 5
##    term                  estimate std.error statistic  p.value
##    <chr>                    <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)            0.0435    0.00514     8.46  3.27e-14
##  2 equipOS-CC            -0.0129    0.00727    -1.77  7.88e- 2
##  3 equipOS-PC             0.00349   0.0112      0.313 7.55e- 1
##  4 equipPAPER            -0.0578    0.0364     -1.59  1.14e- 1
##  5 equipPUNCH            -0.0143    0.0188     -0.759 4.49e- 1
```

```
##  6 econpoor             0.0180     0.00554    3.25   1.45e- 3
##  7 econrich            -0.0157     0.0124    -1.27   2.07e- 1
##  8 usageurban          -0.000674   0.00724   -0.0931 9.26e- 1
##  9 perAA               -0.0390     0.0164    -2.37   1.91e- 2
## 10 equipOS-CC:econpoor -0.0115     0.00971   -1.18   2.41e- 1
## # i 12 more rows
```

- We can use the `drop1()` function to check if a variable can be dropped by the *F-test*, from the *AIC* based model. This can be done to further fine tune the model.

```r
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
```

- We can see that we can drop `usage:perAA` from the model.

```r
finalm <- lm(undercount ~ equip + econ + perAA + equip:econ + equip:perAA, gavote)

tidy_results <- tidy(finalm)

glance_results <- glance(finalm)

print(glance_results)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl>  <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.428         0.359 0.0200      6.20 1.32e-10    17   406. -774. -716.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
print(glance_results)
```


```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl>  <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.428         0.359 0.0200      6.20 1.32e-10    17   406. -774. -716.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

8. Conclusion

let's now attempt an interpretation of this final model. To interpret interactions, it is often helpful to construct predictions for all the levels of the variables involves. Here, we generate all combinations of `equip` and `econ` for a median proportion of `perAA`.

```
median(gavote$perAA)
```

```
## [1] 0.233
```

```
econ <- rep(levels(gavote$econ), 5) # repeat the sequence of middle, poor, rich 5 times

equip <- rep(levels(gavote$equip), rep(3, 5)) # creates a vector that repeats the number 3 five times,

perAA <- 0.233

pdf <- data.frame(econ, equip, perAA)

head(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
```

We now compute the predicted undercount for all 15 combinations and diisplay the result in a table.

```
pred <- predict(finalm, new = pdf)
xtabs(round(pred, 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
```

- `xtabs(... ~ econ + equip, pdf)` creates a contingency table (cross-tabulation) of the rounded predicted values, with econ and equip as the two factors (categorical variables).
- This table summarize the rounded predicted values across the combinations of the levels of `econ` and `equip`, displaying how the predictions are distributed across these two variables.

We can see that the undercount is *lower in richer counties* and *higher in poorer counties*. The amount of difference *depends on the voting system*. Of the three most commonly used voting methods, `LEVER` seems to be the best, offering the lowest proportion of `undercount` for all 3 `econ` classes.

We can use the same appraoch to investigate the relationship between *proportion of African Americans* and the *voting equipment*. We set the proportion of African Americans at 3 levels:

- 1st quartile
- Median
- 3rd quartile

and then we compute the predicted undercount for all types of voting equipment. We set `econ` to only the `middle` level.

```r
# Setup dataframe
econ <- rep("middle", 5); equip <- rep(levels(gavote$equip), rep(3, 5));
perAA <- rep(c(0.11, 0.23, 0.35), 5)
pdf <- data.frame(econ, equip, perAA)
pred2 <- predict(finalm, new = pdf)
```

We now create a 3-level factor for the 3 types of perAA using the `gl()` function. The `gl()` function generates factor levels, with the first argument specifying the number of levels, the second indicating the number of times each level is repeated, and the third defining the total number of observations.

```r
propAA <- gl(n = 3, k = 1, length = 15, labels  = c("low", "medium", "high"))
```

`gl(3, 1, 15, labels = c("low", "medium", "high"))` creates a factor with three levels ("low", "medium", "high") where each level appears once in a sequence until reaching a total of 15 observations. If there are not enough repetitions to meet the total number of observations, the function will recycle the levels.

- `3`: This specifies the number of levels (groups) in the factor. In this case, there will be 3 levels: "low", "medium", and "high".
- `1`: This indicates that each level will be replicated 1 time. So, each level will appear only once in the resulting factor.
- `15`: This is the total number of observations you want to generate. However, in this case, it will only produce 3 unique values (one for each level), and the output will not reach 15.
- `labels = c("low", "medum", "high")`: This provides the labels for the factor levels. The levels of the factor will be labeled as "low", "medium", and "high".

```r
xtabs(round(pred2, 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
```

We see that the effect of the proportion of African Americans on undercount is mixed. High proportions are predicted to be associated with higher levels of the OS-CC voting method, while low proportions are predicted to be associated with lower undercounts for LEVER and PUNCH. Additionally, low proportions of African Americans are associated with negative proportions of PAPER voting methods, which could indicate *that low paper-based voting methods may reduce undercount in counties that have low proportions of African Americans and also rank low in terms of socioeconomic status and literacy rates.*

In summary, we have found that the *economic status* of a county is the *clearest factor determining the proportion of undercount votes*, with richer counties having lower undercounts. The type of equipment used and the proportion of African Americans do have some impact on undercount, but the direction of the effect cannot be estimated in this analysis.

9. Libraries Used

```r
library(tidyverse)
library(visreg)
library(faraway)
library(viridis)
library(RColorBrewer)
```