Regression Models Course Project - Kevin O'Leary

Executive summary

Using linear regression models, this report looks at the questions whether manual or automatic transmission cars have higher mpg, and how big the difference is. Mpg of manual transmission cars is significantly higher: by 7.2 miles. This result is qualified by controlling for weight: the difference between manual and automatic transmission cars is smaller for heavier cars.

Questions

This report tries to answer two questions, based on the mtcars dataset (n=32):

1. Is an automatic transmission better for MPG?
2. Quantify the MPG difference between automatic and manual transmissions.

Exploratory analysis

All variables correlate significantly with `mpg`, between 0.42 and 0.87 (results omitted for brevity).

```
##         cyl          disp         hp           drat        wt
## corr    -0.852162    -0.8475514   -0.7761684   0.6811719   -0.8676594
## p-value 6.112687e-10 9.380327e-10 1.787835e-07 1.77624e-05 1.293959e-10
##         qsec       vs           am           gear        carb
## corr    0.418684   0.6640389    0.5998324    0.4802848   -0.5509251
## p-value 0.01708199 3.415937e-05 0.0002850207 0.005400948 0.001084446
```

To find variables that should be adjusted for, I used chi-square tests to check the relationship between the categorial variables and `am`. It turns out that the variables `cyl` and `gear` differ depending on `am` (results omitted for brevity).

For the numerical variables: `disp`, `drat`, and `wt` differ depending on `am` (based on t-tests, results omitted for brevity).

Figure 1 shows the relationship between `mpg` and `am` both depending on `cyl` and `gear`. Figure 2 to 4 show the relationships between `mpg` and `am` depending on `dips`, `drat`, and `wt`.

Regression results: manual transmission cars have better mpg than automatic transmission cars

The boxplots (see figure 1) suggest that manual transmission cars have higher mpg values than automatic transmission cars. A simple linear regression of `mpg` on `am` confirms this:

```
## $coefficients
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 17.147368   1.124603 15.247492 1.133983e-15
## ammanual     7.244939   1.764422  4.106127 2.850207e-04
```

The mean of mpg for the **manual transmission cars is 7.2 miles higher than for the automatical transmission cars**, a significant difference (p=0.0003).

With 95% confidence we can estimate that the difference between automatic and manual transmission cars is between 3.65 and 10.85 miles per gallon.

```
##                    lower bound upper bound
## mean automatic        14.85062    19.44411
## manual - automatic     3.64151    10.84837
```

The residuals of the simple regression seem to be random and normally distributed: see figure 5.

No data point has an overly strong influence:

```
range(hatvalues(fit))
```

```
## [1] 0.05263158 0.07692308
```

Car weight is related to both transmission and mpg. Looking at the three plots relating `disp`, `drat`, and `wt` to `mpg` (see fig. 2 to 4), the difference between automatic and manual transmission seemed to be the greatest in the case of weight. So let's adjust for weight:

```
## $coefficients
##                Estimate Std. Error     t value     Pr(>|t|)
## (Intercept) 37.32155131  3.0546385 12.21799285 5.843477e-13
## ammanual    -0.02361522  1.5456453 -0.01527855 9.879146e-01
## wt          -5.35281145  0.7882438 -6.79080719 1.867415e-07
```

This model explains much more variance than the model that includes transmission alone (75% as opposed to 36%). Adjusting for weight, we see that difference in transmission disappears.

Let's include the interaction term:

```
## $coefficients
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 31.416055  3.0201093 10.402291 4.001043e-11
## ammanual    14.878423  4.2640422  3.489277 1.621034e-03
## wt          -3.785908  0.7856478 -4.818836 4.551182e-05
## ammanual:wt -5.298360  1.4446993 -3.667449 1.017148e-03
```

Now we see that there is in fact a difference between automatic and manual transmission: the mpg of both manual and automatic transmission cars drops the heavier the car, but the mpg drop is much steeper for manual than for automatic transmission cars.
Concretely:

- if the weight of an automatic transmission car goes up by 1000lbs, the expected mpg drops by roughly 3.8 miles.
- if the weight of a manual transmission car goes up by 1000lbs, the expected mpg drops only roughly 9.1 miles – more than twice as fast.

This model with the interaction explains 83% of variance (as opposed to 75% for the model without the interaction term).

Based on the relationships established in the exploratory section, `cyl` have an influence on both `mpg` and transmission. Including cylinders to the interaction (`mpg ~ am*wt + cyl`) makes a significant, but small difference: it adds only 5% explained variance. This small additional value of including cylinders is confirmed by an anova between the models.

Conclusions

Using linear regression models, we tried to answer whether manual or automatic transmission cars have higher mpg, and how big the difference is.

A single variable linear regression shows that the mpg of manual transmission cars is with 95% confidence between 3.6 and 10.8 miles higher.

This result is qualified by controlling for weight: the difference in between manual and automatic transmission cars is smaller for heavier cars.

Limitations

Not included in this report are the influences of other variables of the dataset. Some exploratory results suggested that they wouldn't change the relationship of transmission and mpg greatly, but there is some room for further research.

Applying regression to variables like number of cylinders violate the assumption of normality, so the results are somewhat questionable. Still, the visual analysis (looking at the plots) corroborates the regression results.

Appendix: Figures

Figure 1: Relationship between `mpg` and `am`, depending on `cyl` and `gear`:

```
## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.
```

```
## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.
```

```
## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.
```

```
## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.

## Warning in FUN(X[[32L]], ...): EXPR is a "factor", treated as integer.
##  Consider using 'switch(as.character( * ), ...)' instead.
```

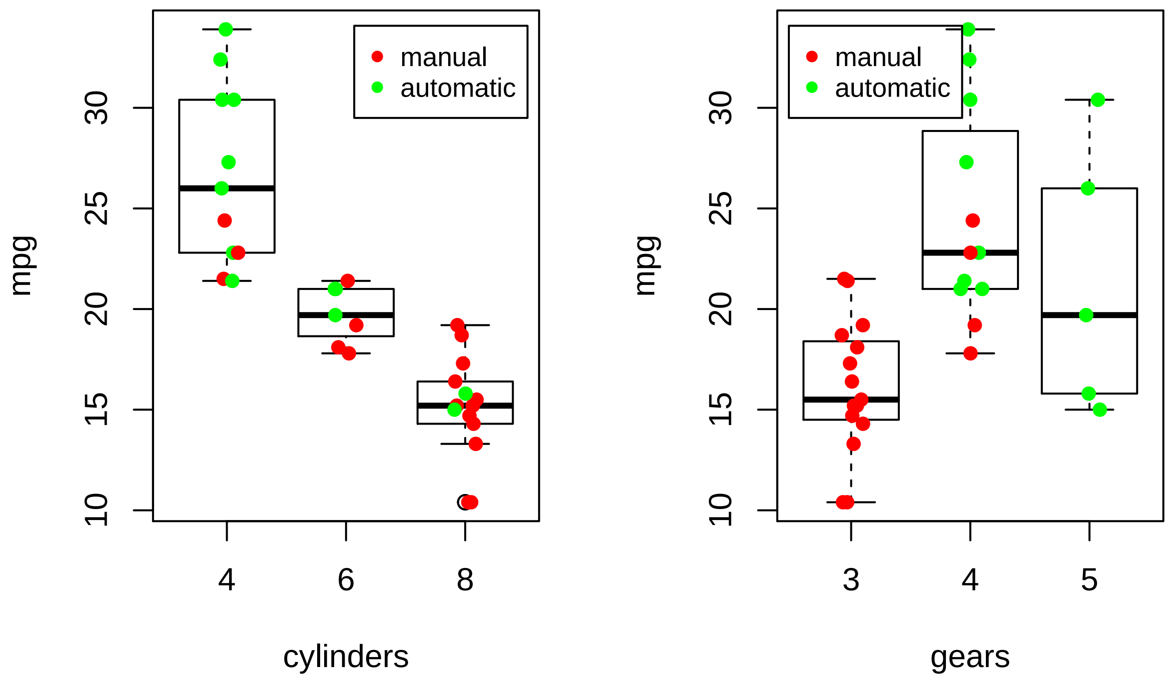

Figure 2: Relationship between `mpg` and `disp` separated by `am`:

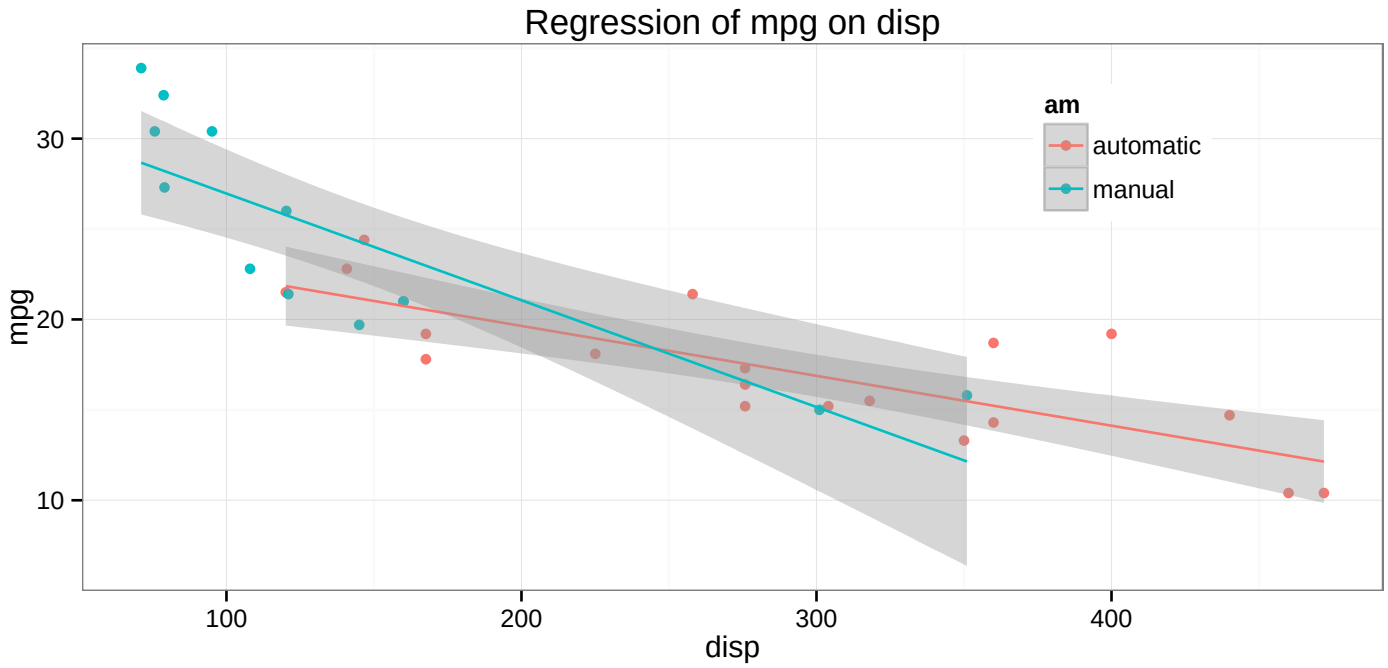

Figure 3: Relationship between `mpg` and `drat` separated by `am`:

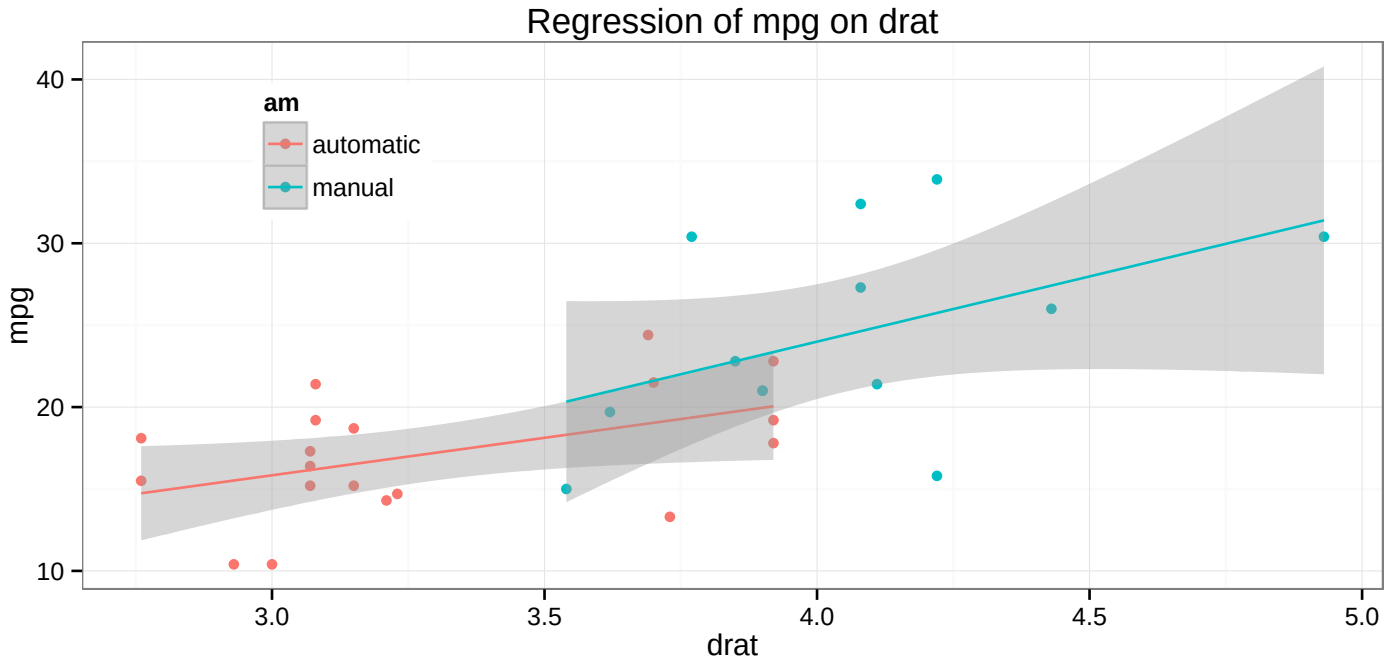

Figure 4: Relationship between `mpg` and `wt` separated by `am`:

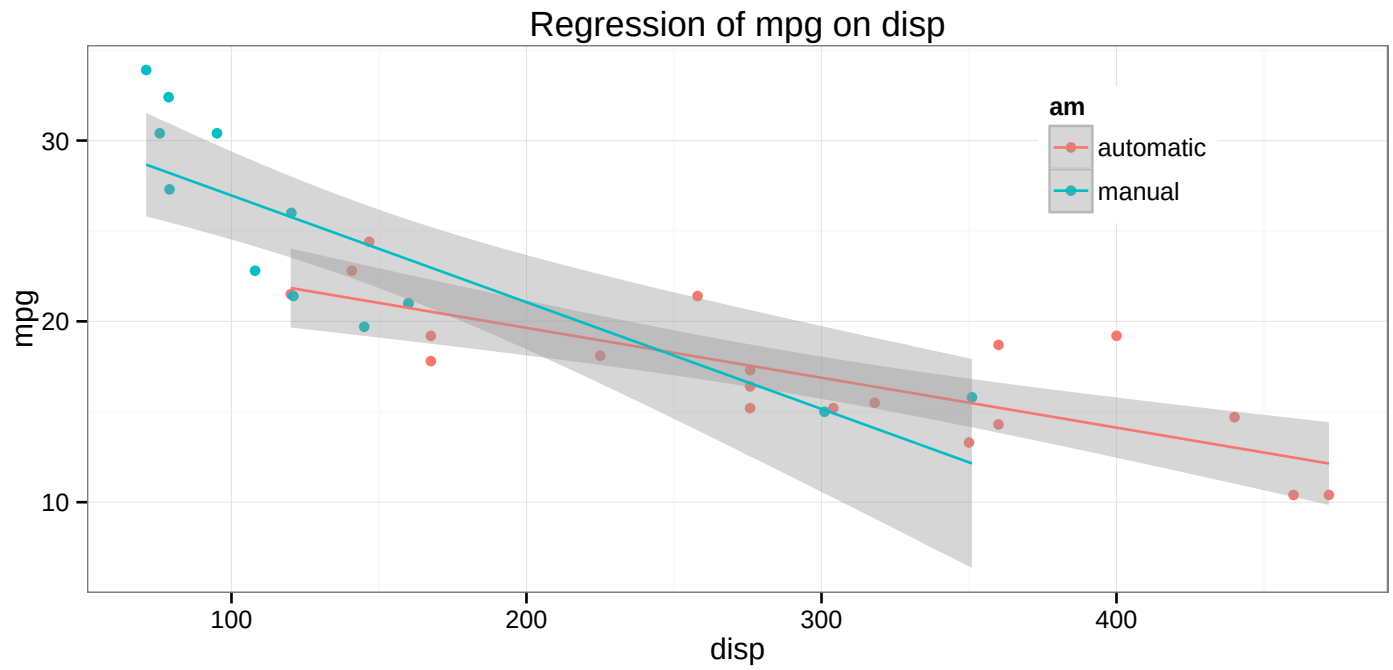

Figure 5: Residual plot and plot of distribution for model `mpg ~ am`: