Transformation of SLR

Cody Frisby

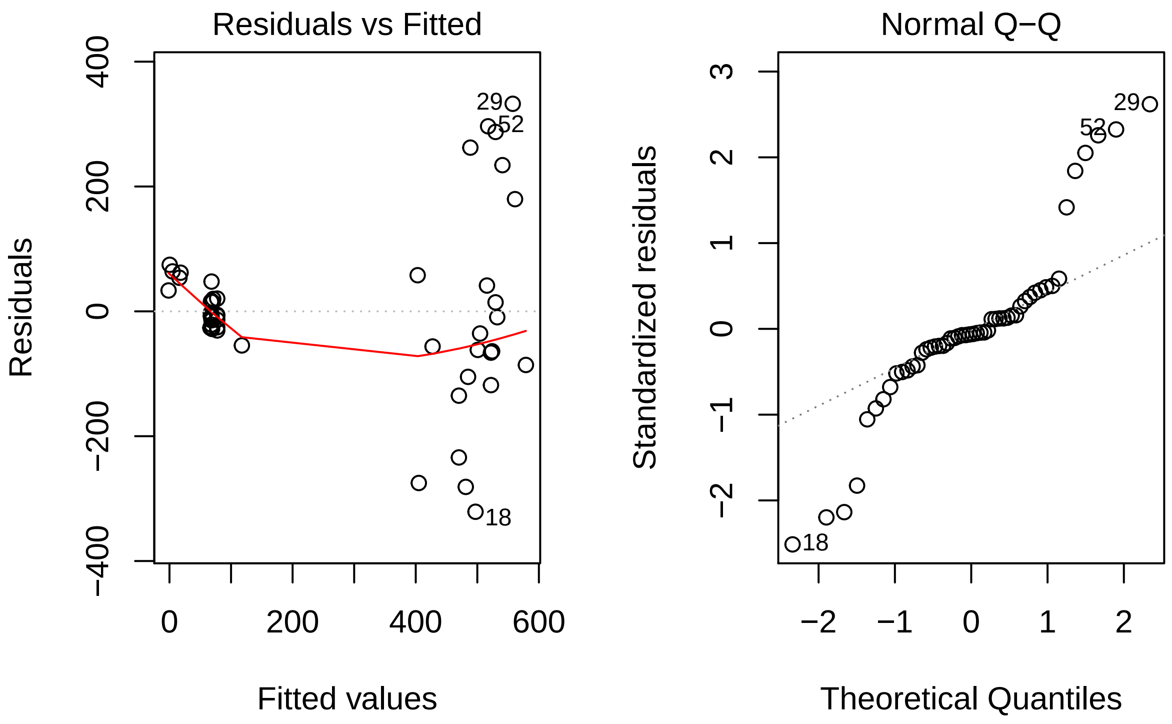
February 12, 2016

1) Prediction or Relationship?

Prediction is most important here. If we can predict sales of beer based on the price, with minimal error, our budgets, inventory, and even sales could be planned accordingly.

2) Fitting a SLR

From inspection of the residual plots of the linear model $sales = 1812.1839386 + (-93.0072726)price$ we cannot assume linearity, equality of variances, or normality. There are many outliers on both plots and a definite funneling shape to the residuals by fitted plot, meaning as the fitted value increases so do the residuals. We do not have homoscedasticity of variances.



3) Possible Transformation

The mathematical form of our transformed model is

$$y_i^{\lambda} = \beta_0 + \beta_1 x_i + \varepsilon_i$$

where the transformation applied is raising the y_i^{th} observation to the power of λ (lambda). Lambda is just a number whose value is obtained through a more complex analysis of our fitted linear model. B_0 is the intercept of our transformed model, B_1 is the slope of our transformed model and ε are the errors from our transformed model. This transformation of y will allow us to reduce the error around our original linear model and we should be able to better predict sales given a price.

4)

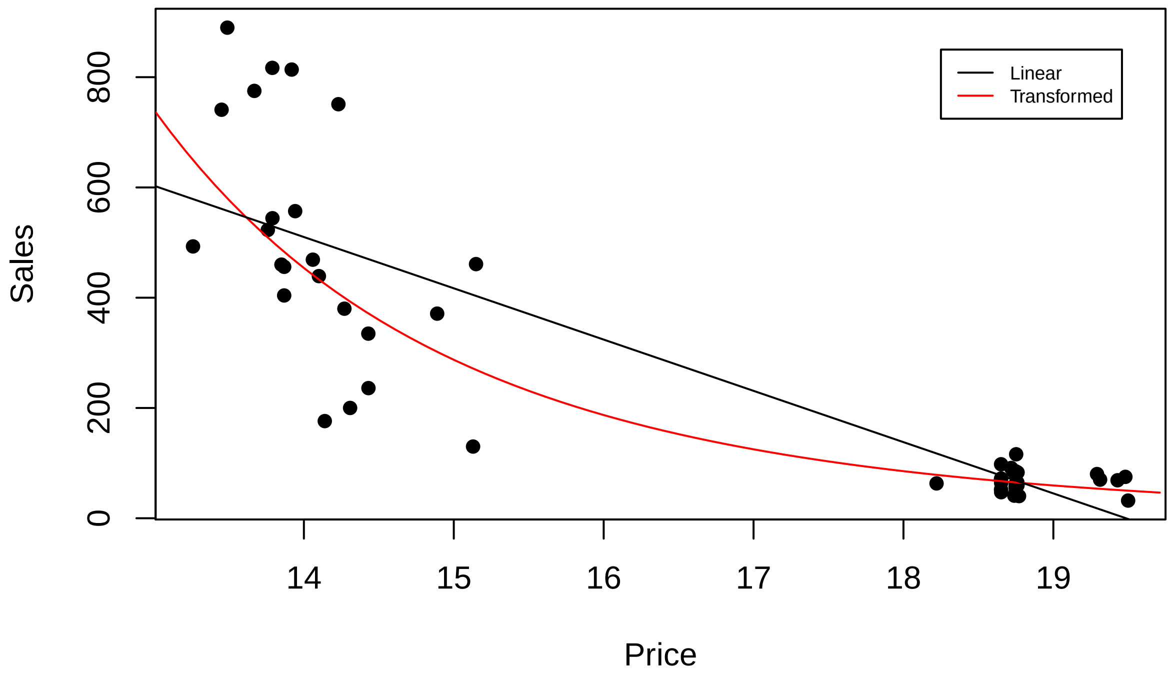
The model was fit by raising y to -0.144773 and then fitting a linear model to this data. Our transformed model is

$$sales^{-0.144773} = 0.0173989 + 0.0282163 price$$

But this isn't super intuitive. We can raise both sides to $\frac{1}{-0.144773}$ to make the output of the transformed data be in the units we care about, cases of beer sold or predicted cases of beer sold.

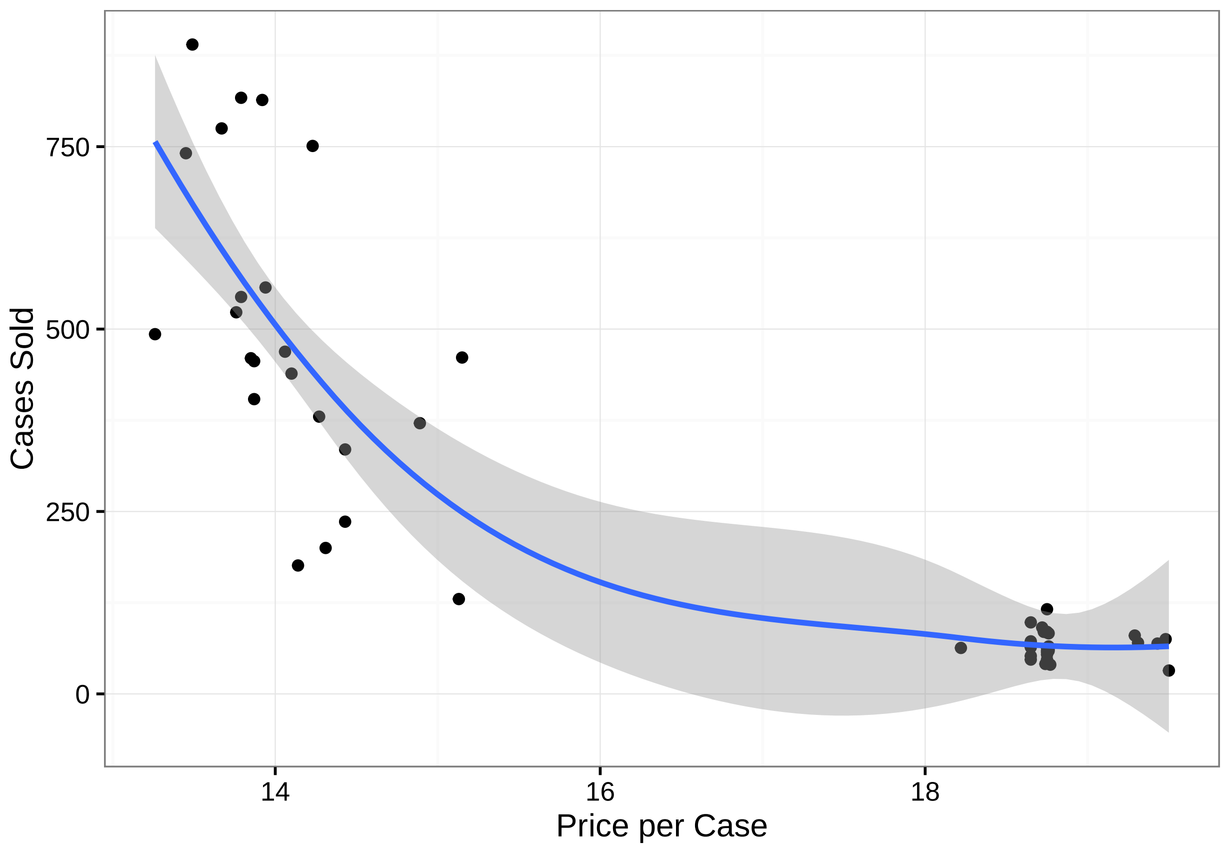
$$sales = (0.0173989 + 0.0282163 price)^{\frac{1}{-0.144773}}$$

And here we plot our transformed model onto the original scatter plot.

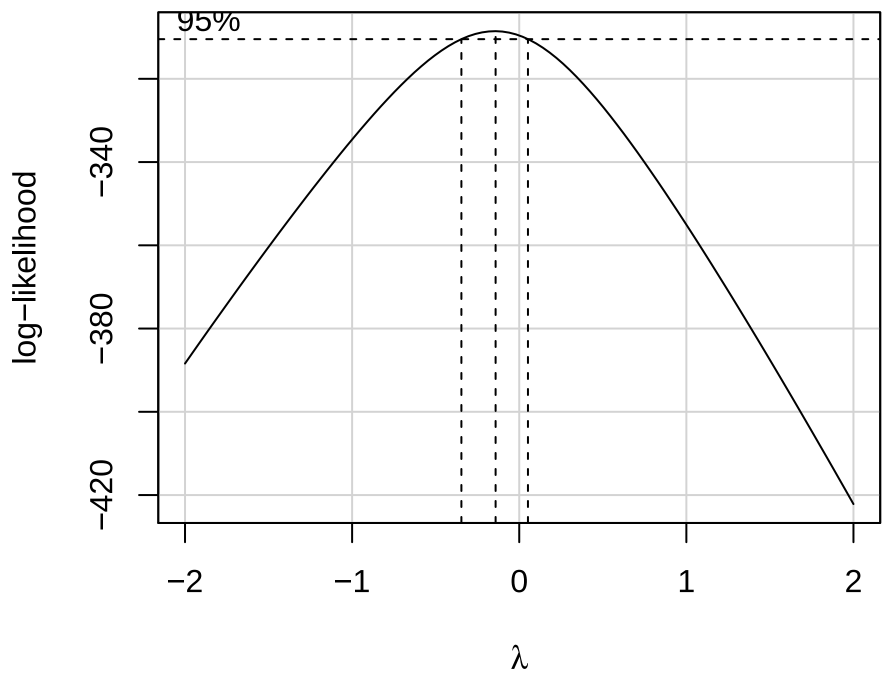


5)

Fitting a smoother line to the data gave me an initial clue as to what model that data may best fit. Here I fit the data using a "loess" method in R. It looks as though a possible transformation could be a negative exponential.

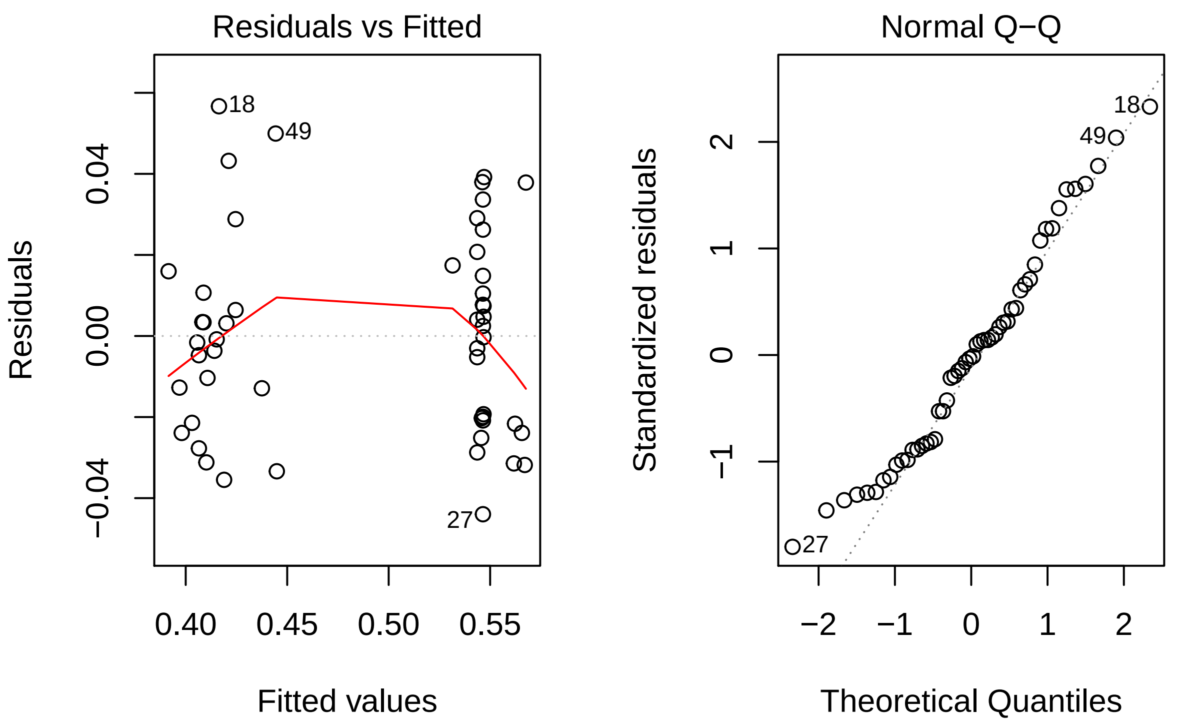


Also, a box cox plot gave me clue about what power of lambda I might raise the data to for a possible transformation.



Based on the powerTransform R function the value for $\lambda = -0.144773$.

Here we look at the residual plots from this transformed model.



I think we can now safely assume that the assumptions of linearity, equality of variance, and normality hold. Here's a snap shot of our data with an added column for the transformed predicted values.

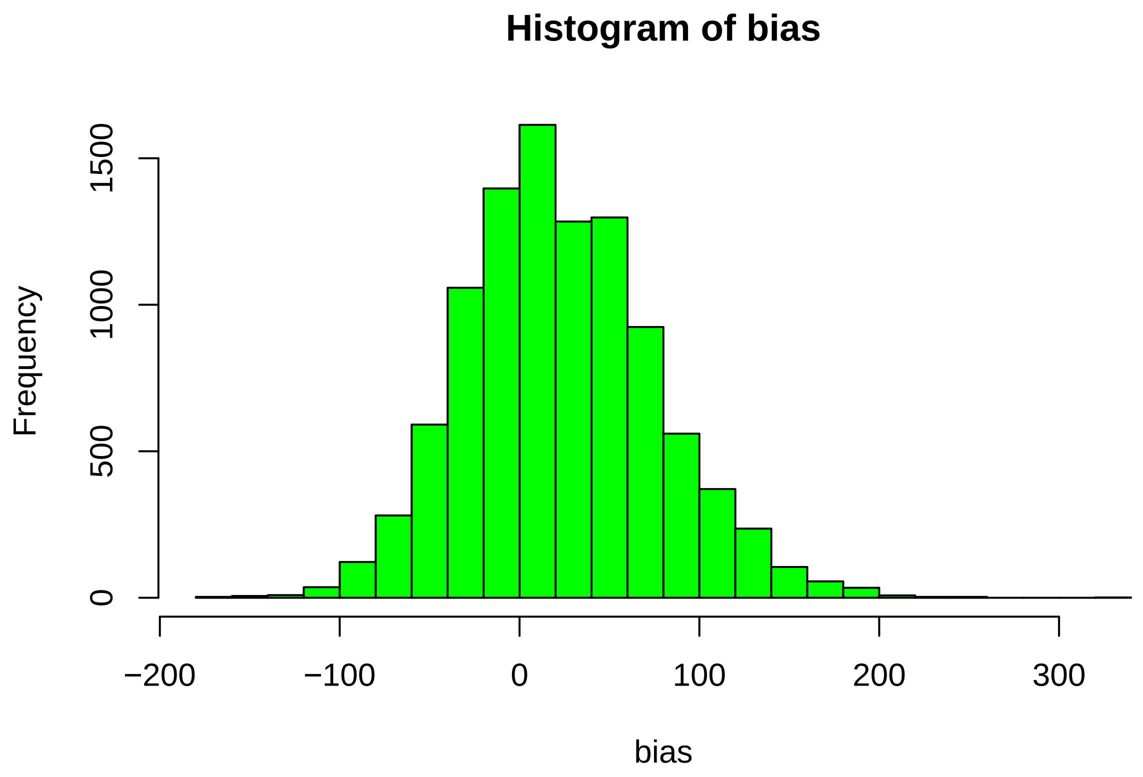
Week	PRICE.18PK	CASES.18PK	pred.transform
1	14.10	439	432.99813
2	18.65	98	67.35069
3	18.65	70	67.35069
4	18.65	52	67.35069
5	18.65	64	67.35069
6	18.65	72	67.35069

The SSE (sum squared residuals) from the original linear model were 8.5188482×10^5. The SSE from the transformed model were 7.4765278×10^5. The value for R^2 is 0.7811655 from the transformed model. This means that we are able to explain approximately 78% of the variation in beer sales just by knowing the price of beer. This is very useful and allows us to make predictions with less error.

Now, I need to perform a cross validation of our transformed model by taking a simple random sample from the original data, fit the transformed model, and then with the remaining, unsampled, data test this model.

predicted_bias	mean_square_bias	rpmse
40.8888	29378.44	171.4014

Keep in mind that this is only one of many tests performed on our transformation. We could get a better idea of the true bias of our model if we were to run this simulation over and over.



Running this simulation 10,000 times gives us, on average, a bias of 22.7699114. So, we may be in danger of overestimating sales by 20+ cases, on average, using this model. A 95% inverval for this simulation yields $[-72.751202, 135.4135818]$.

6) Predicted Sales

A)

How we would compute the predictions by hand and have them in the correct units.

$$sales = (0.0173989 + 0.0282163 * price)^{\frac{1}{-0.144773}}$$

If we price beer at \$15/case we would predict to sell 287.4 cases. If we price beer at \$14/case we would predict to sell 453.9 cases. We predict that we would sell 454 - 287 = 167 more cases of beer moving the price from 15 to 14 dollars.

B) 95% confidence and prediction intervals when price is \$14

	lower	upper
confidence	381.5371	542.3587
prediction	202.8963	1129.2942

Here is a table showing the prediction and confidence intervals.

7) Conclusions

Even though we do not have a linear relationship between sales and price, applying the appropriate transformation allows to still be able to build a useful model. I think this model will be able to predict as well as any, particularly better than assuming a linear one. However, there is much variability in the lower price ranges which will make it difficult to be precise in our prediction accuracy. There may be other "noise" that we do not have explanation for at this point. I may recommend looking into this if more accuracy is desired.

R code:

```
# block 1
# read the data aand fit a linear model
# library(car) contains some functions I use below.
library(car)
beer <- read.csv("~/Documents/MATH3710/ProblemSets/problem3/beer.csv")
beer <- subset(beer, complete.cases(beer))
x <- beer$PRICE.18PK
y <- beer$CASES.18PK
fit <- lm(y ~ x)
b0fit <- fit$coefficients[1]
b1fit <- fit$coefficients[2]
# block 2
par(mfrow = c(1,2))
plot(fit, which = c(1,2))
# block 3
# note, we can use the function powerTransform(model).
lam <- powerTransform(fit)
lam <- lam$lambda
fit4 <- lm(y^lam ~ x)
# to create a table we can display some of the data if desired.
b0t <- fit4$coefficients[1]
b1t <- fit4$coefficients[2]
E <- cbind(x, y, y^lam)
colnames(E) <- c("Price", "Cases", "Cases^lam")
# block 4
# plot the transformed model onto the original scatter plot.
plot(jitter(y)~jitter(x), xlab = "Price", ylab = "Sales", pch=16)
lines(seq(par()$usr[1], par()$usr[2], 0.1),   (fit4$coefficients[1]+fit4$coefficients[2]*seq(par()$usr[
legend(17, 850, "y = (0.017 + 0.028x)^(-1/0.144773)", lty = 1, col = 2, cex=0.6)
# block 5
# try to make better looking plots with ggplot2
library(ggplot2)
g <- ggplot(beer, aes(x = PRICE.18PK, y = CASES.18PK))
g <- g + geom_point()
g <- g + stat_smooth(method = "loess", formula = y ~ x)
g <- g + theme_bw()
g <- g + xlab("Price per Case") + ylab("Cases Sold")
g
# block 6
boxCox(fit) # to see what power of lambda may be best for the transformation.
# block 7
par(mfrow = c(1,2))
```

```r
plot(fit4, which = c(1,2))
# block 8
Est <- coef(fit)
SE <- coef(summary(fit))[,2]
Est.t <- coef(fit4)
SE.t <- coef(summary(fit4))[,2]
B <- matrix(c(cbind(Est, SE),cbind(Est.t, SE.t)), 4, 2)
colnames(B) <- c("y ~ x", "y^lam ~ x")
rownames(B) <- c("Intercept", "x", "SE(intercept)", "SE(x)")
knitr::kable(B)
# block 9, predict using the transformed model
test <- data.frame(x = c(15,14))
beer.conf <- predict(fit4, newdata = test, se.fit = TRUE,
                        interval = "confidence")
beer.predict <- predict(fit4, newdata = test, se.fit = TRUE,
                        interval = "prediction")
beer.p.b <- beer.conf$fit^(1/lam)
differ <- beer.p.b[2,1] - beer.p.b[1,1]
se.15 <- beer.conf$se.fit[1]
se.14 <- beer.conf$se.fit[2]
bias <- vector()
# test and cross validate our model
for (i in 1:10000) {
  ind <- sample(1:52, 6)
  # now we can subset the carmpg data frame into 2 data frames using this index.
  test <- beer[beer$index %in% ind, ]
  training <- beer[!(beer$index %in% ind), ]
  # now to fit a new model with our subsetted training data.
  fit.train <- lm(CASES.18PK^lam ~ PRICE.18PK, data = training)
  testing <- predict.lm(fit.train, newdata = test)
  # here is the predicted bias:
  predicted_bias <- mean(test$CASES.18PK - (testing)^(1/lam))
  bias <- rbind(bias, predicted_bias)
}
hist(bias, col = "green", breaks = 20)
# block 10
beer.conf <- beer.conf$fit^(1/lam)
beer.pre <- beer.predict$fit^(1/lam)
A <- rbind(beer.conf[2,3:2], beer.pre[2,3:2])
rownames(A) <- c("confidence", "prediction")
colnames(A) <- c("lower", "upper")
knitr::kable(A)
```