hw6

Mike Lehman

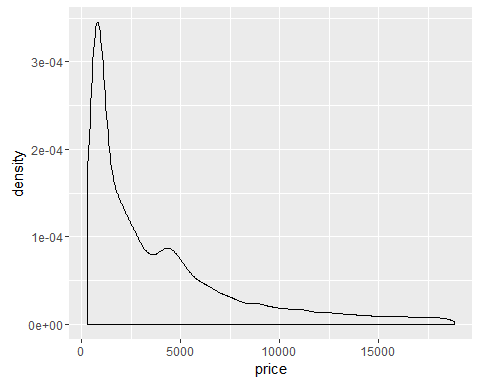
October 11, 2016

## #1

library(ggplot2)  
data(diamonds)  
  
price\_df <- as.data.frame(diamonds[,"price"])  
colnames(price\_df) <- c("price")  
summary(price\_df$price)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 326 950 2401 3933 5324 18820

d <- ggplot(price\_df["price"], aes(x = price))  
d + geom\_density()

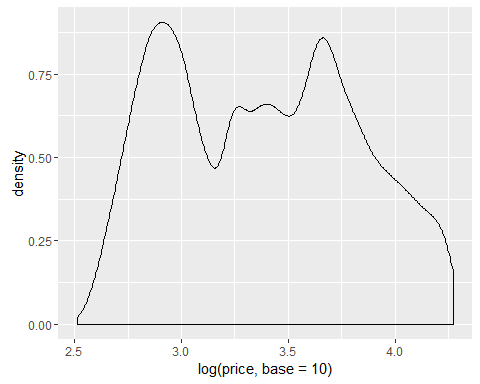


## #2

d\_log10 <- ggplot(price\_df["price"], aes(x = log(price, base = 10)))  
summary(log(price\_df$price, base = 10))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2.513 2.978 3.380 3.382 3.726 4.275

d\_log10 + geom\_density()



Performing a log (base 10) transformation on the price variable was worthwhile in this case because it reduced the scope of the varialbe's distribution and brought the mean and median much closer than they were in the raw data.

Since the range of the raw data for the price variable was so wide and all values were positive, a log transformation was worthwhile.

## #3

set.seed(123)  
train\_sample <- sample(53940, 26970)  
  
diamonds\_train <- diamonds[train\_sample, ]  
diamonds\_test <- diamonds[-train\_sample, ]  
  
str(diamonds\_train)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 26970 obs. of 10 variables:  
## $ carat : num 1.03 0.5 1.74 0.51 0.7 0.71 0.32 0.9 0.32 1.52 ...  
## $ cut : Ord.factor w/ 5 levels "Fair"<"Good"<..: 4 1 3 5 5 5 3 1 5 4 ...  
## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 4 1 5 1 5 5 2 4 4 4 ...  
## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 4 3 2 4 3 8 3 1 6 4 ...  
## $ depth : num 62.3 65.7 62.1 61.2 62.5 62 63.3 65.1 61.6 62.6 ...  
## $ table : num 59 56 59 55 56 54 55 58 56 55 ...  
## $ price : int 6214 1323 10086 1882 2294 3190 672 1939 708 12958 ...  
## $ x : num 6.38 5.01 7.65 5.18 5.64 5.71 4.36 6.05 4.42 7.39 ...  
## $ y : num 6.42 4.97 7.78 5.16 5.69 5.75 4.33 5.99 4.44 7.28 ...  
## $ z : num 3.99 3.28 4.79 3.16 3.54 3.55 2.75 3.92 2.73 4.59 ...

str(diamonds\_test)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 26970 obs. of 10 variables:  
## $ carat : num 0.21 0.24 0.22 0.3 0.2 0.32 0.3 0.3 0.23 0.23 ...  
## $ cut : Ord.factor w/ 5 levels "Fair"<"Good"<..: 4 3 1 2 4 4 2 3 3 3 ...  
## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 6 2 7 2 2 7 7 2 5 ...  
## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 3 7 4 3 2 1 3 3 4 5 ...  
## $ depth : num 59.8 62.3 65.1 64 60.2 60.9 63.4 62.7 63.8 61 ...  
## $ table : num 61 57 61 55 62 58 54 59 55 57 ...  
## $ price : int 326 336 337 339 345 345 351 351 352 353 ...  
## $ x : num 3.89 3.95 3.87 4.25 3.79 4.38 4.23 4.21 3.85 3.94 ...  
## $ y : num 3.84 3.98 3.78 4.28 3.75 4.42 4.29 4.27 3.92 3.96 ...  
## $ z : num 2.31 2.47 2.49 2.73 2.27 2.68 2.7 2.66 2.48 2.41 ...

## #4

price\_reg <- lm(price ~ carat + cut + color + clarity, data = diamonds\_train)  
summary(price\_reg)

##   
## Call:  
## lm(formula = price ~ carat + cut + color + clarity, data = diamonds\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16783.6 -676.0 -199.6 464.9 9734.9   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -3724.458 19.872 -187.427 < 2e-16 \*\*\*  
## carat 8894.155 17.143 518.809 < 2e-16 \*\*\*  
## cut.L 720.433 28.599 25.191 < 2e-16 \*\*\*  
## cut.Q -339.831 25.209 -13.481 < 2e-16 \*\*\*  
## cut.C 175.185 21.832 8.024 1.06e-15 \*\*\*  
## cut^4 -4.072 17.523 -0.232 0.816   
## color.L -1928.960 25.072 -76.936 < 2e-16 \*\*\*  
## color.Q -632.759 22.835 -27.710 < 2e-16 \*\*\*  
## color.C -200.682 21.366 -9.393 < 2e-16 \*\*\*  
## color^4 17.243 19.642 0.878 0.380   
## color^5 -91.521 18.592 -4.923 8.59e-07 \*\*\*  
## color^6 -42.536 16.930 -2.512 0.012 \*   
## clarity.L 4268.663 43.956 97.112 < 2e-16 \*\*\*  
## clarity.Q -1794.710 41.145 -43.619 < 2e-16 \*\*\*  
## clarity.C 966.876 35.192 27.474 < 2e-16 \*\*\*  
## clarity^4 -311.778 28.153 -11.075 < 2e-16 \*\*\*  
## clarity^5 213.314 22.962 9.290 < 2e-16 \*\*\*  
## clarity^6 6.386 19.977 0.320 0.749   
## clarity^7 108.414 17.631 6.149 7.90e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1162 on 26951 degrees of freedom  
## Multiple R-squared: 0.915, Adjusted R-squared: 0.915   
## F-statistic: 1.612e+04 on 18 and 26951 DF, p-value: < 2.2e-16

price\_reg\_log10 <- lm(log(price, base = 10) ~ carat + cut + color + clarity, data = diamonds\_train)  
summary(price\_reg\_log10)

##   
## Call:  
## lm(formula = log(price, base = 10) ~ carat + cut + color + clarity,   
## data = diamonds\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.59629 -0.09512 0.02553 0.10781 0.72075   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.5712714 0.0025103 1024.273 < 2e-16 \*\*\*  
## carat 0.9550545 0.0021657 440.991 < 2e-16 \*\*\*  
## cut.L 0.0280646 0.0036129 7.768 8.26e-15 \*\*\*  
## cut.Q -0.0104373 0.0031846 -3.277 0.00105 \*\*   
## cut.C 0.0110580 0.0027580 4.009 6.10e-05 \*\*\*  
## cut^4 0.0006581 0.0022137 0.297 0.76625   
## color.L -0.2156317 0.0031673 -68.080 < 2e-16 \*\*\*  
## color.Q -0.0662756 0.0028847 -22.975 < 2e-16 \*\*\*  
## color.C 0.0011813 0.0026991 0.438 0.66162   
## color^4 0.0203386 0.0024814 8.196 2.58e-16 \*\*\*  
## color^5 -0.0061295 0.0023487 -2.610 0.00906 \*\*   
## color^6 0.0004805 0.0021388 0.225 0.82224   
## clarity.L 0.3364601 0.0055529 60.592 < 2e-16 \*\*\*  
## clarity.Q -0.1613721 0.0051978 -31.046 < 2e-16 \*\*\*  
## clarity.C 0.0996837 0.0044458 22.422 < 2e-16 \*\*\*  
## clarity^4 -0.0288661 0.0035565 -8.116 5.00e-16 \*\*\*  
## clarity^5 0.0243339 0.0029008 8.389 < 2e-16 \*\*\*  
## clarity^6 -0.0007686 0.0025237 -0.305 0.76069   
## clarity^7 0.0028658 0.0022273 1.287 0.19823   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1468 on 26951 degrees of freedom  
## Multiple R-squared: 0.8887, Adjusted R-squared: 0.8886   
## F-statistic: 1.196e+04 on 18 and 26951 DF, p-value: < 2.2e-16

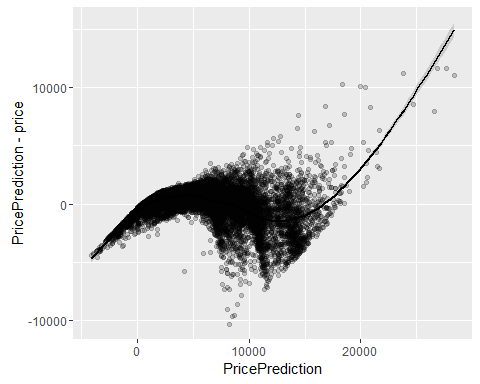
## #5

diamonds\_test$price\_log10 <- log(diamonds\_test$price, base = 10)  
diamonds\_test$PricePrediction <- predict(price\_reg, newdata = diamonds\_test)  
diamonds\_test$PricePrediction\_log10 <- predict(price\_reg\_log10, newdata = diamonds\_test)  
head(diamonds\_test)

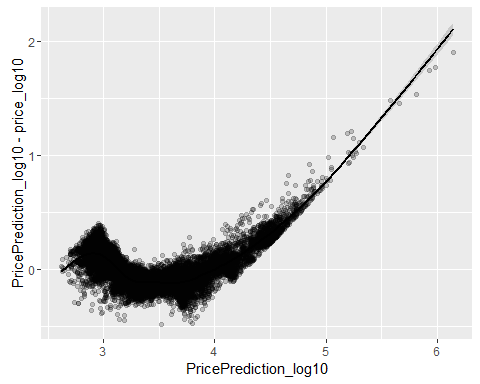
## carat cut color clarity depth table price x y z  
## 2 0.21 Premium E SI1 59.8 61 326 3.89 3.84 2.31  
## 7 0.24 Very Good I VVS1 62.3 57 336 3.95 3.98 2.47  
## 9 0.22 Fair E VS2 65.1 61 337 3.87 3.78 2.49  
## 11 0.30 Good J SI1 64.0 55 339 4.25 4.28 2.73  
## 15 0.20 Premium E SI2 60.2 62 345 3.79 3.75 2.27  
## 16 0.32 Premium E I1 60.9 58 345 4.38 4.42 2.68  
## price\_log10 PricePrediction PricePrediction\_log10  
## 2 2.513218 -1272.3089 2.839535  
## 7 2.526339 -765.5969 2.805364  
## 9 2.527630 -1432.2482 2.860746  
## 11 2.530200 -2839.7210 2.700325  
## 15 2.537819 -2338.2799 2.750487  
## 16 2.537819 -3875.1446 2.625297

## #6

a <- ggplot(data = diamonds\_test, aes(x = PricePrediction, y = PricePrediction - price))  
a + geom\_point(alpha = 0.2, color = "black") + geom\_smooth(aes(x = PricePrediction, y = PricePrediction - price), color = "black")



b <- ggplot(data = diamonds\_test, aes(x = PricePrediction\_log10, y = PricePrediction\_log10 - price\_log10))  
b + geom\_point(alpha = 0.2, color = "black") + geom\_smooth(aes(x = PricePrediction\_log10, y = PricePrediction\_log10 - price\_log10), color = "black")

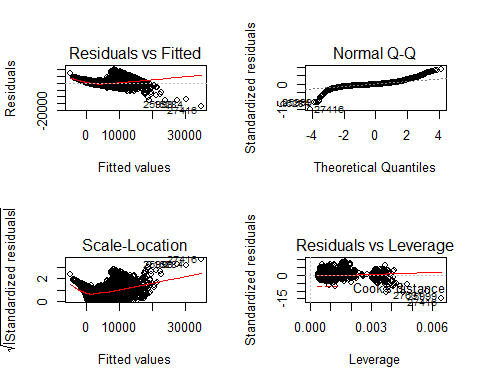


## #7

confint(price\_reg)

## 2.5 % 97.5 %  
## (Intercept) -3763.40701 -3685.508642  
## carat 8860.55272 8927.756634  
## cut.L 664.37808 776.488675  
## cut.Q -389.24142 -290.420219  
## cut.C 132.39330 217.977114  
## cut^4 -38.41811 30.274047  
## color.L -1978.10257 -1879.817296  
## color.Q -677.51636 -588.001163  
## color.C -242.55950 -158.803804  
## color^4 -21.25723 55.742617  
## color^5 -127.96121 -55.080188  
## color^6 -75.71984 -9.351879  
## clarity.L 4182.50736 4354.819504  
## clarity.Q -1875.35720 -1714.063640  
## clarity.C 897.89719 1035.854204  
## clarity^4 -366.95901 -256.597615  
## clarity^5 168.30644 258.321130  
## clarity^6 -32.77062 45.542074  
## clarity^7 73.85660 142.972047

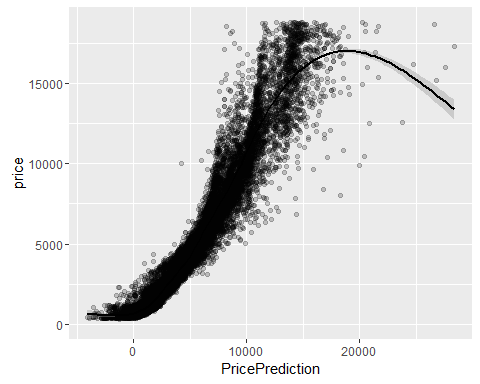
par(mfrow=c(2,2))  
plot(price\_reg)



c <- ggplot(data = diamonds\_test, aes(x = PricePrediction, y = price))  
c + geom\_point(alpha = 0.2, color = "black") + geom\_smooth(aes(x = PricePrediction, y = price), color = "black") + geom\_line(aes(x = log(PricePrediction, base = 10), y = log(price, base = 10)), color = "blue", linetype = 2)

## Warning: NaNs produced

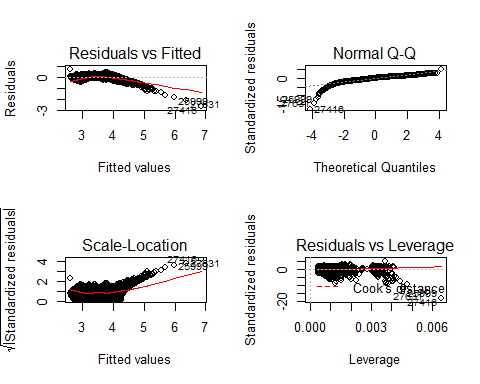
## Warning: Removed 2550 rows containing missing values (geom\_path).



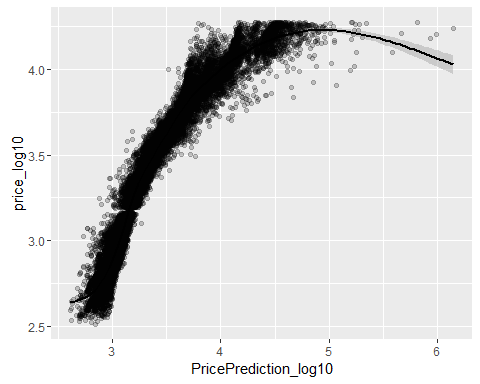
confint(price\_reg\_log10)

## 2.5 % 97.5 %  
## (Intercept) 2.566351056 2.576191843  
## carat 0.950809583 0.959299355  
## cut.L 0.020983220 0.035145987  
## cut.Q -0.016679233 -0.004195295  
## cut.C 0.005652153 0.016463831  
## cut^4 -0.003680786 0.004996994  
## color.L -0.221839817 -0.209423581  
## color.Q -0.071929765 -0.060621441  
## color.C -0.004109033 0.006471702  
## color^4 0.015474948 0.025202226  
## color^5 -0.010732976 -0.001526022  
## color^6 -0.003711588 0.004672580  
## clarity.L 0.325576159 0.347344100  
## clarity.Q -0.171560117 -0.151184138  
## clarity.C 0.090969756 0.108397663  
## clarity^4 -0.035836994 -0.021895200  
## clarity^5 0.018648231 0.030019655  
## clarity^6 -0.005715213 0.004177915  
## clarity^7 -0.001499871 0.007231382

par(mfrow=c(2,2))  
plot(price\_reg\_log10)



d <- ggplot(data = diamonds\_test, aes(x = PricePrediction\_log10, y = price\_log10))  
d + geom\_point(alpha = 0.2, color = "black") + geom\_smooth(aes(x = PricePrediction\_log10, y = price\_log10), color = "black")



From the above graphs and plots, we can see that both models do an acceptable job of predicting the price.

Looking at the confidence intervals for each model, it appears that carat, cut, and clarity are the most strongly correlated with the dependent variable, price. Both models also seem to have a somewhat linear pattern. There is not perfect linearity here, but the lines in the above graphs show general linearity and the values are largely consistent among the scatter.

If choosing which model is more accurate, the log(base 10) transformed model yielded better results between the two. This is likely because transforming the price variable brought the range of values closer together, and the median a better resprentative of a specific candidate.