model_building

J. Wall

March 5, 2019

Steps in Model Building

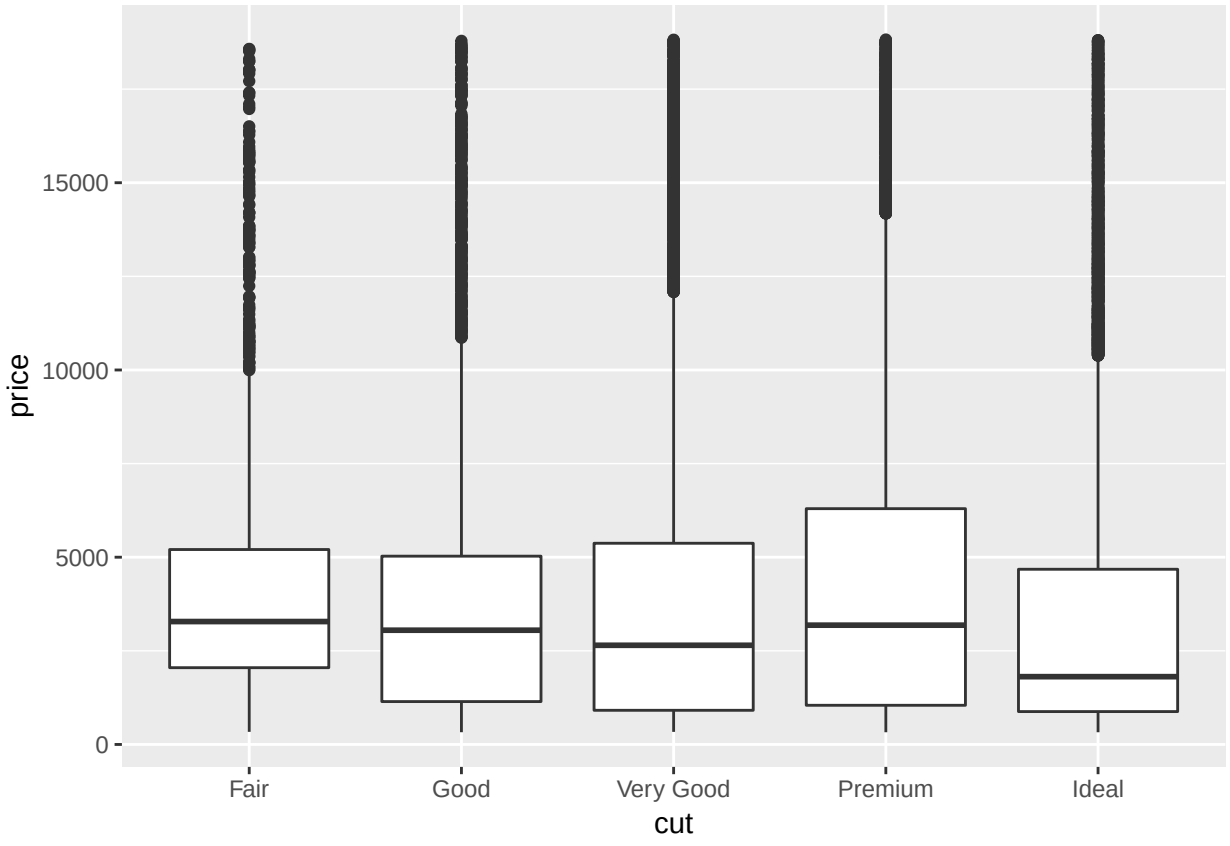
- Split data into training, query and test data
- Visualize data
- Transform if needed
- Model the data and partition data into pattern and residuals
- repeat the process using residuals instead of old response variable

Pros and cons between model building and a machine learning approach: machine learning generally has better predictive capability, but is frequently a black box that lacks the ability to apply real-world knowledge to the problem or to interpret the model.

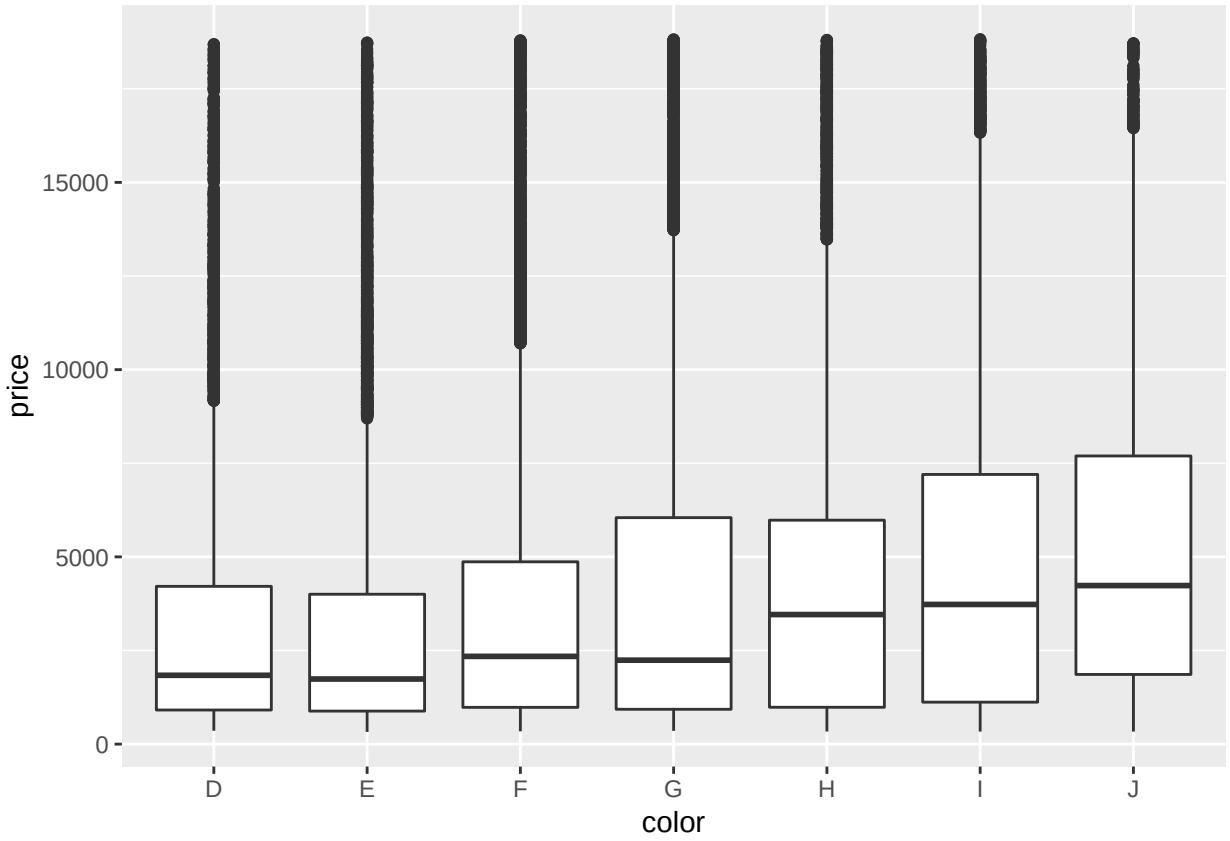
Diamonds dataset application

At first glance, it would seem that lower quality diamonds are more expensive. Look at cut, color and clarity as predictors of price. Note that the worst color is J and the worst claristy is I1.

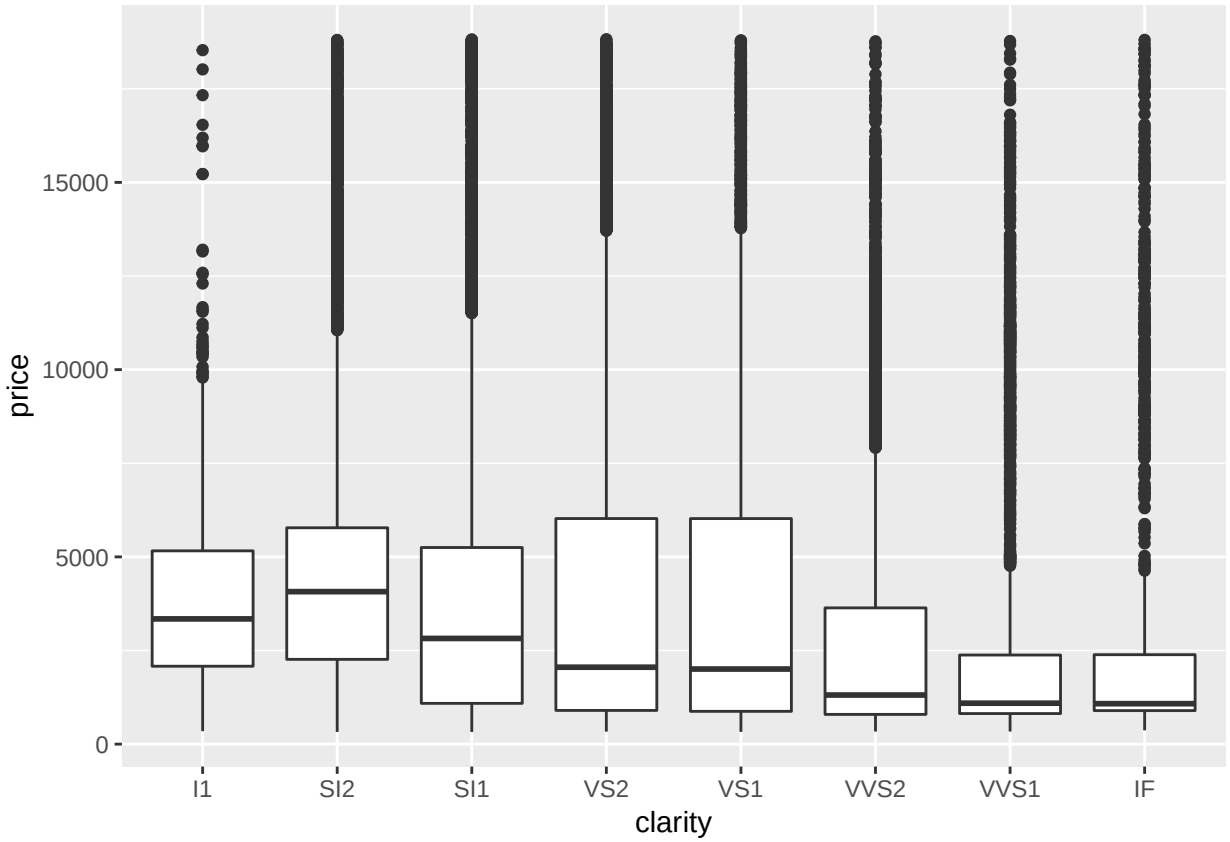
```
ggplot(diamonds, aes(cut, price)) + geom_boxplot()
```




```
ggplot(diamonds, aes(color, price)) + geom_boxplot()
```

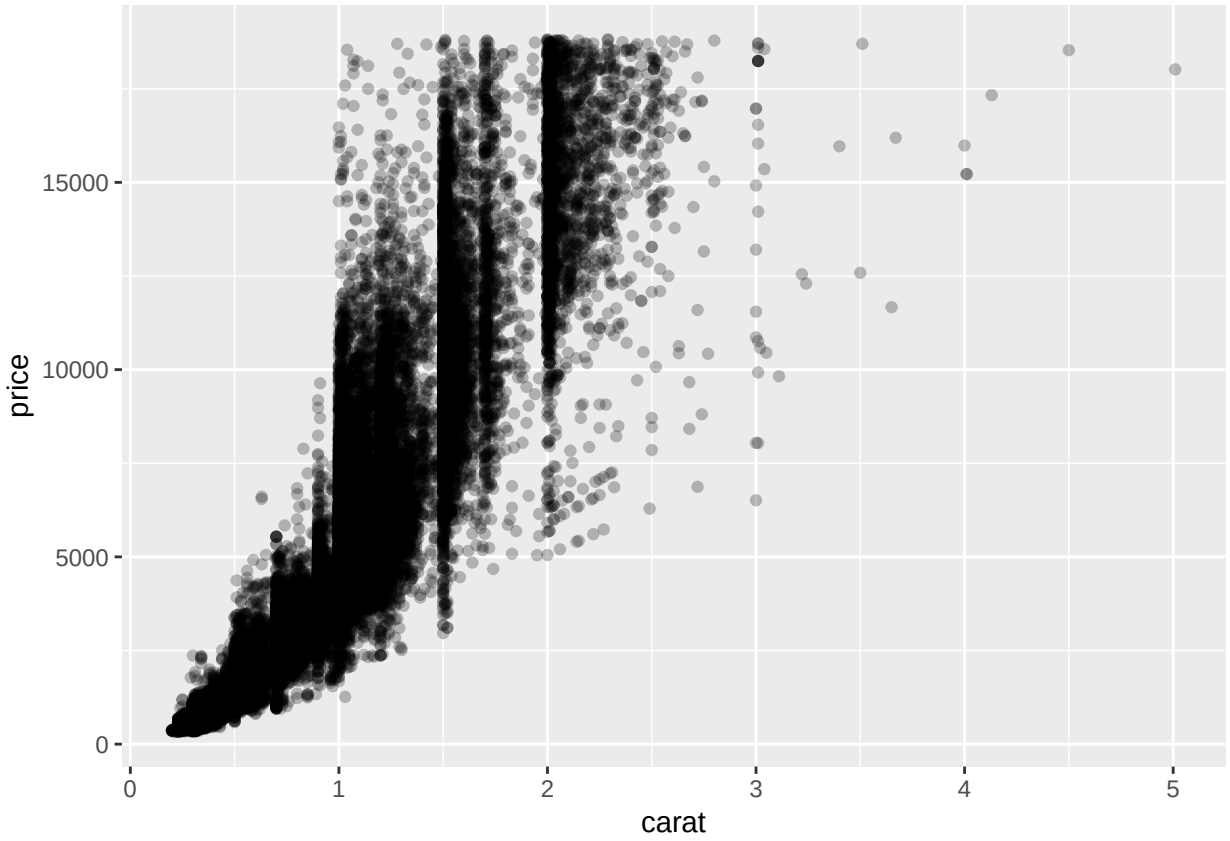



```
ggplot(diamonds, aes(clarity, price)) + geom_boxplot()
```

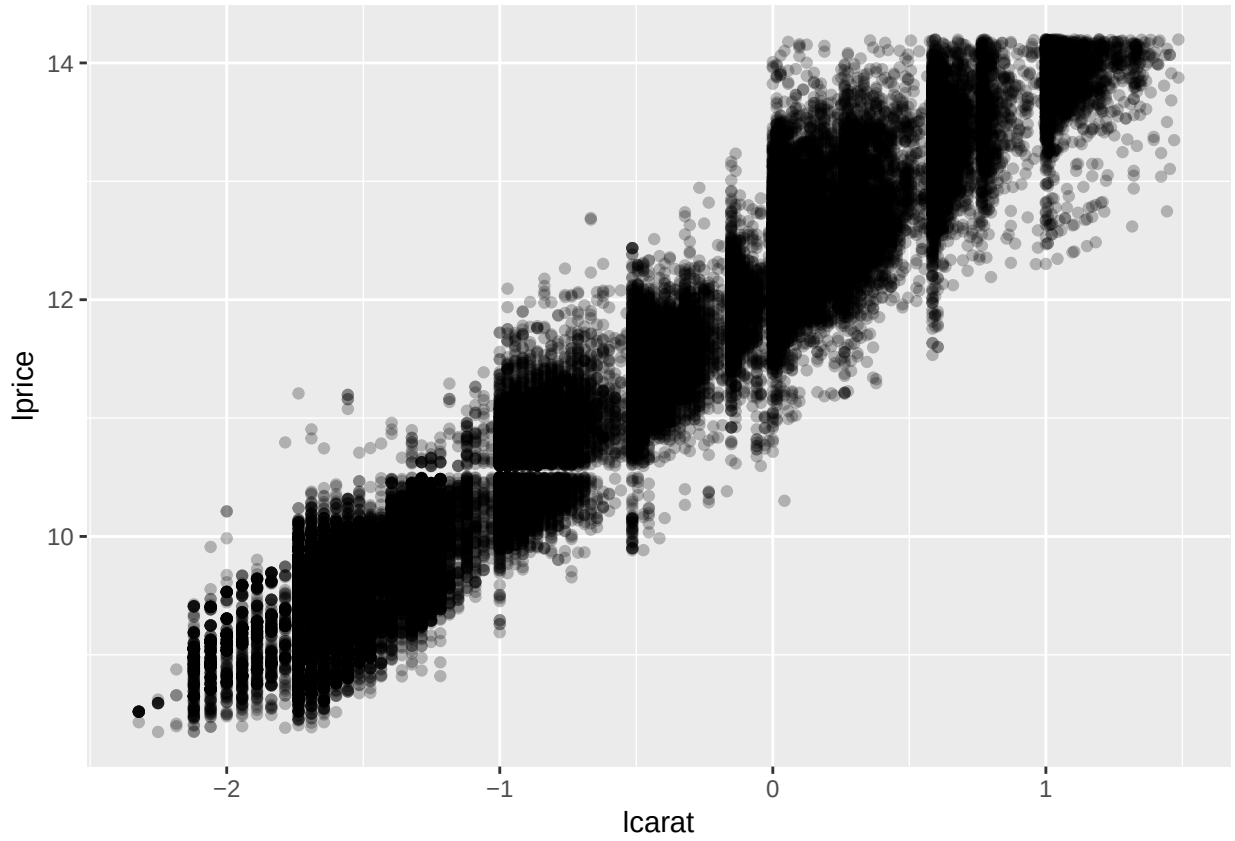


What could be causing this? Perhaps we have not found yet the main thing effecting price. Look at carat and its effect on price.

```
ggplot(diamonds, aes(carat, price)) +
  geom_point(alpha = 0.25)
```

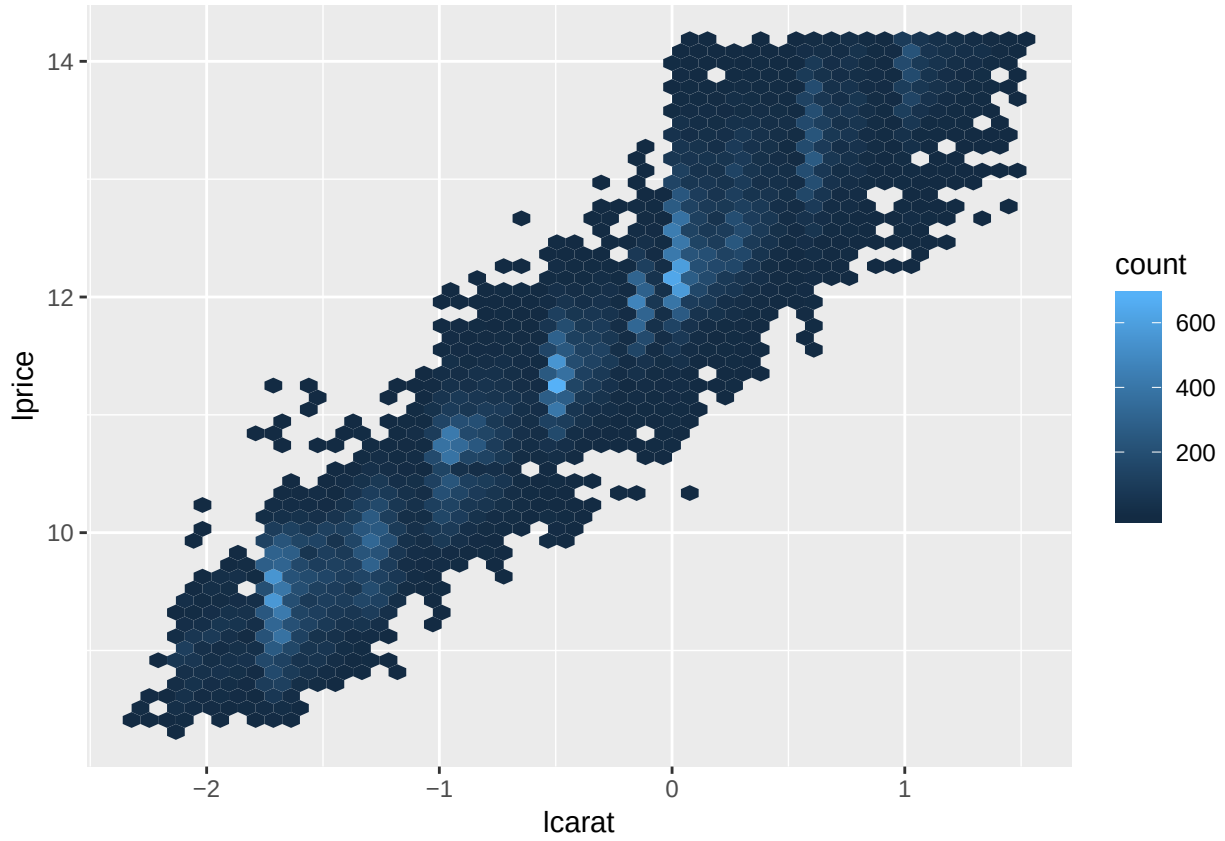



```
# let's look at doing a log transform of the data and leaving off the really large diamonds
diamonds2 <- diamonds %>%
  filter(carat <= 2.8) %>%
  mutate(lprice = log2(price), lcarat = log2(carat))
# now visualize the transformed dataset
ggplot(diamonds2, aes(lcarat, lprice)) +
   geom_point(alpha = 0.25)
```




```
ggplot(diamonds2, aes(lcarat, lprice)) +
   geom_hex(bins = 50)
```

```
## Warning: package 'hexbin' was built under R version 3.5.3
```



Now we model and remove the linear pattern. Then we will look at the residuals.

```
mod_diamonds <- lm(lprice ~ lcarat, data = diamonds2)
x <- c(5,3,4,6,5,4.5)
seq_range(x, 10)
```

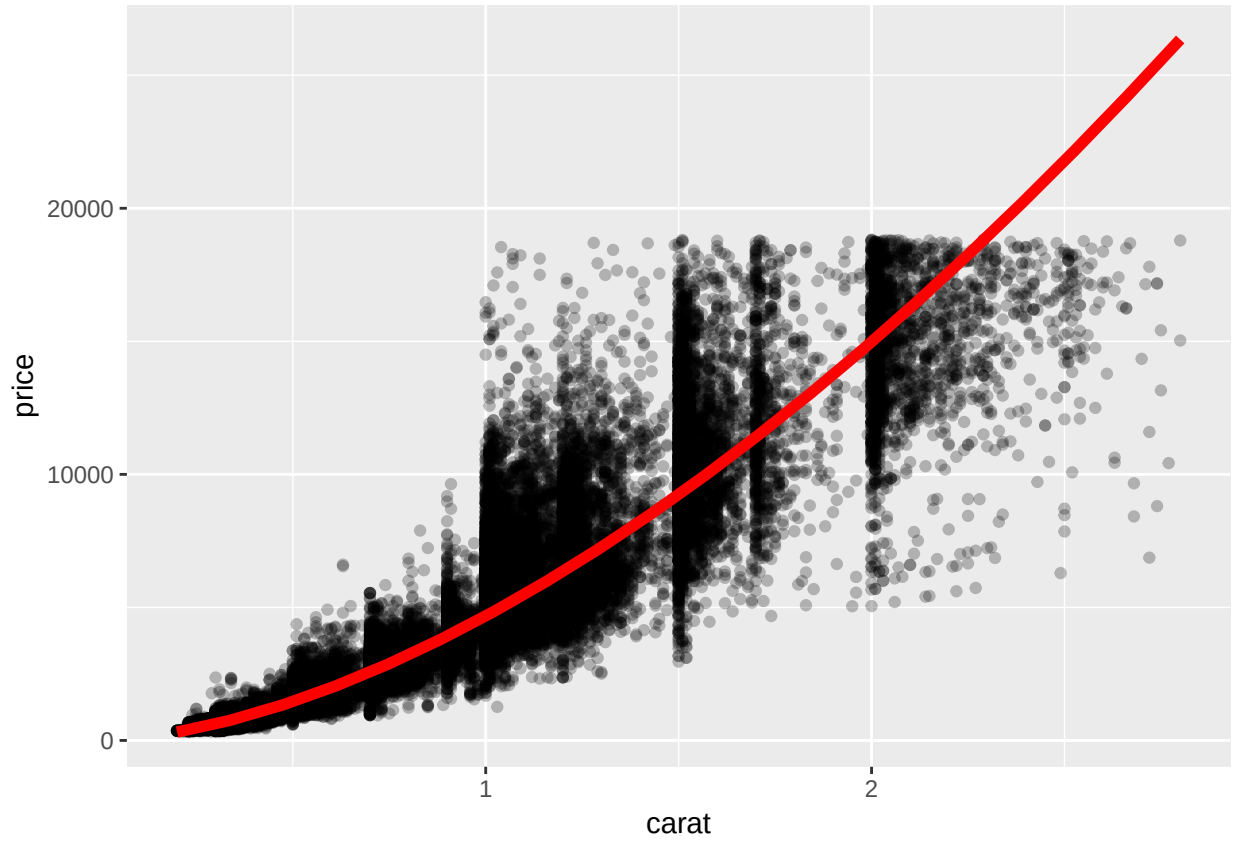
```
##  [1] 3.000000 3.333333 3.666667 4.000000 4.333333 4.666667 5.000000
##  [8] 5.333333 5.666667 6.000000
```

```
# set up a grid with predicted values for range of carat
grid <- diamonds2 %>%
  data_grid(carat = seq_range(carat, 20)) %>%
  mutate(lcarat = log2(carat)) %>%
  add_predictions(mod_diamonds, "lprice") %>%
  mutate(price = 2 ^ lprice)
grid
```

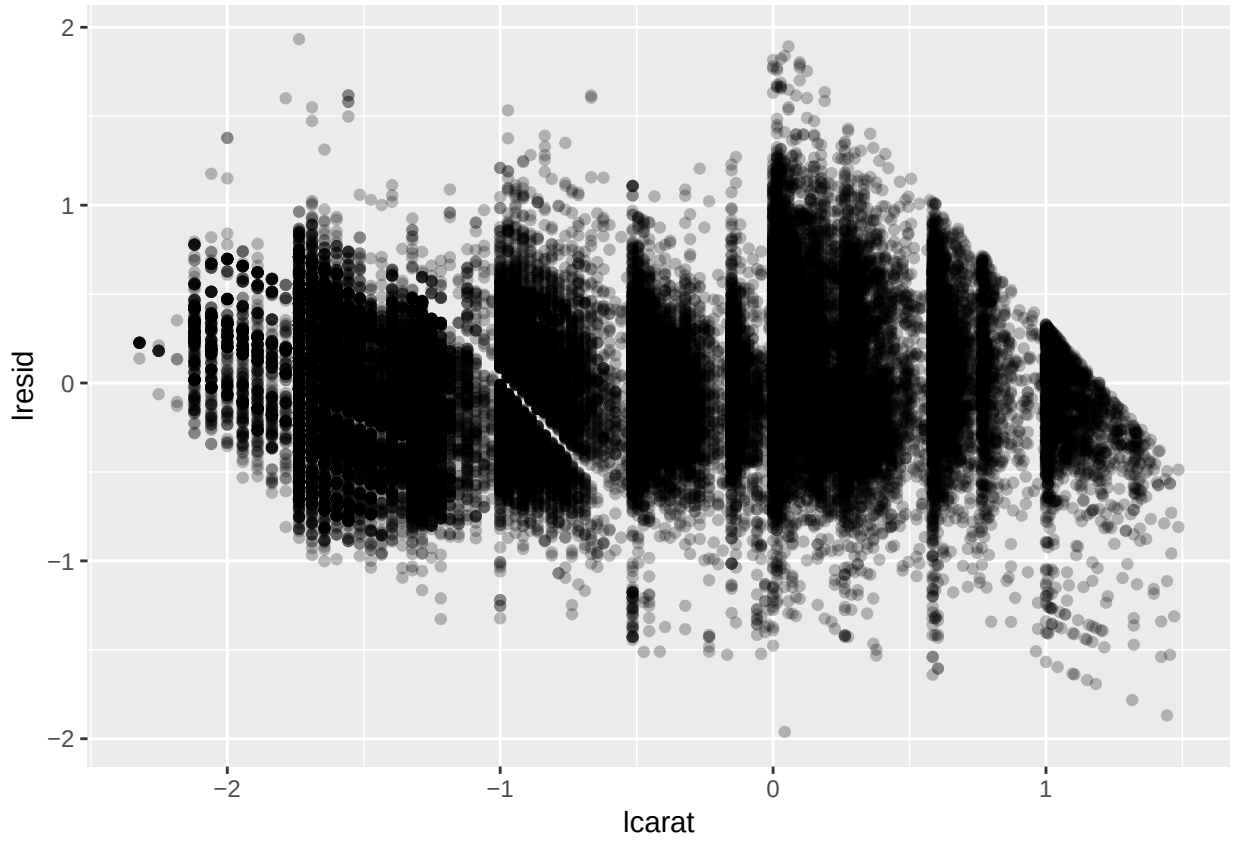
```
## # A tibble: 20 x 4
##    carat  lcarat lprice  price
##    <dbl>   <dbl>  <dbl>  <dbl>
##  1 0.2   -2.32     8.29   314.
##  2 0.337 -1.57     9.56   753.
##  3 0.474 -1.08    10.4   1334.
##  4 0.611 -0.712   11.0   2043.
##  5 0.747 -0.420   11.5   2869.
##  6 0.884 -0.178   11.9   3804.
##  7 1.02   0.0301  12.2   4844.
##  8 1.16   0.212   12.5   5983.
```

```
##  9 1.29   0.373   12.8   7217.
## 10 1.43   0.518   13.1   8543.
## 11 1.57   0.649   13.3   9958.
## 12 1.71   0.770   13.5  11459.
## 13 1.84   0.881   13.7  13044.
## 14 1.98   0.985   13.8  14712.
## 15 2.12   1.08    14.0  16460.
## 16 2.25   1.17    14.2  18286.
## 17 2.39   1.26    14.3  20189.
## 18 2.53   1.34    14.4  22167.
## 19 2.66   1.41    14.6  24220.
## 20 2.8    1.49    14.7  26346.
```

```r
# plot original points and prediction line
ggplot(diamonds2, aes(carat, price)) +
  geom_point(alpha = 0.25) +
  geom_line(data = grid, color = "red", size = 2)
```

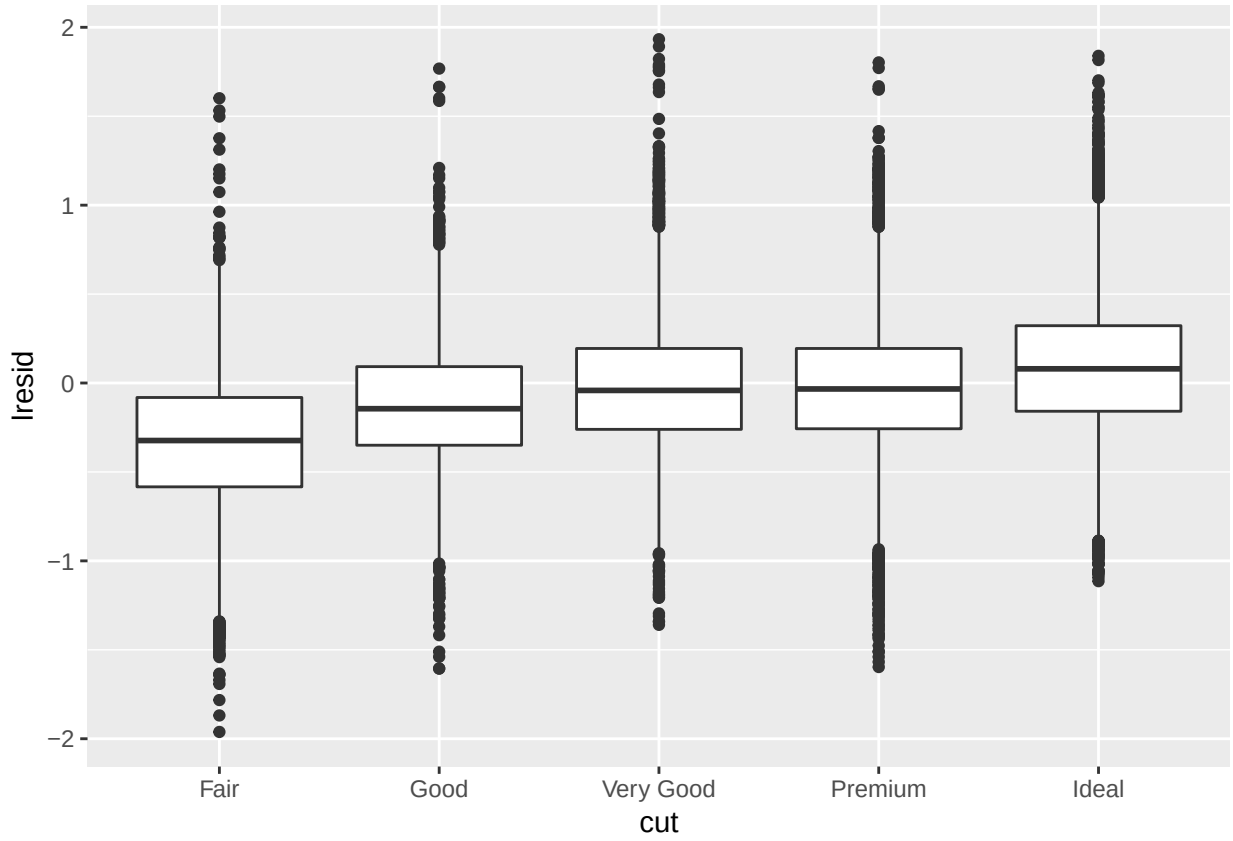



```r
#  add residuals to the diamonds dataset
diamonds2 <- diamonds2 %>%
  add_residuals(mod_diamonds,"lresid")
diamonds2 %>% ggplot(aes(lcarat, lresid)) +
  geom_point(alpha = 0.25)
```

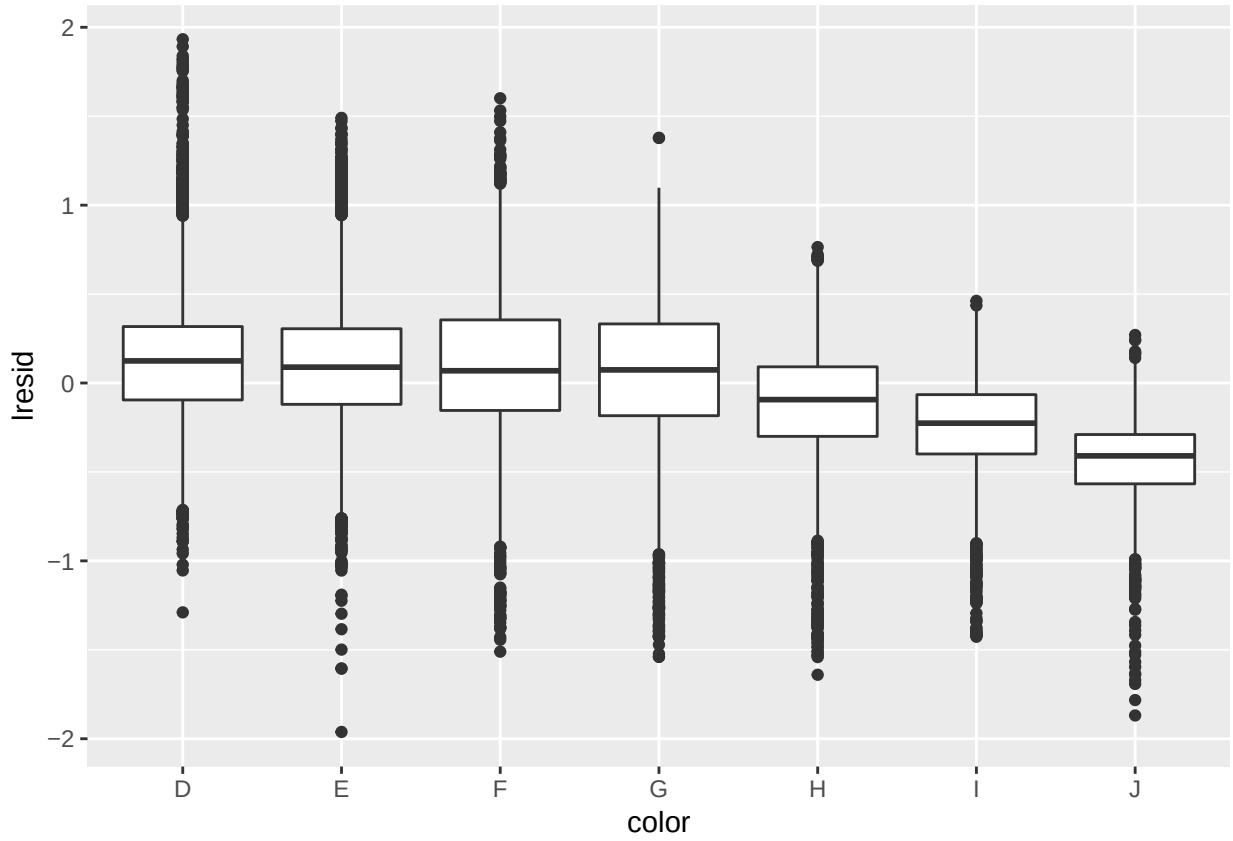


Now we can re-do our initial plots using the residuals instead of the price. Note that a residual of -1 says that lprice is 1 unit lower than the prediction based solely on the number of carats. So, it is 2^(-1) or 1/2 the price expected solely on its weight.

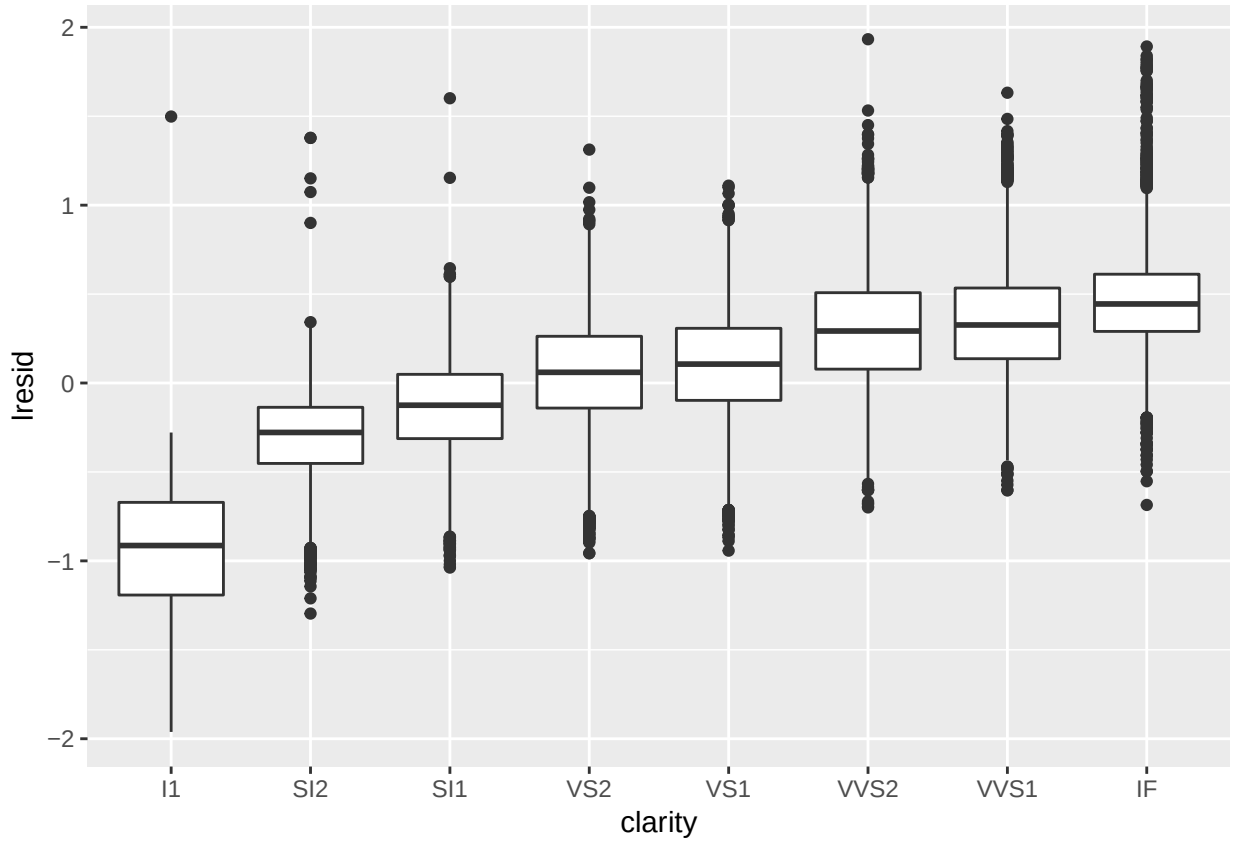
```
ggplot(diamonds2, aes(cut, lresid)) + geom_boxplot()
```




```
ggplot(diamonds2, aes(color, lresid)) + geom_boxplot()
```




```
ggplot(diamonds2, aes(clarity, lresid)) + geom_boxplot()
```



More complicated model

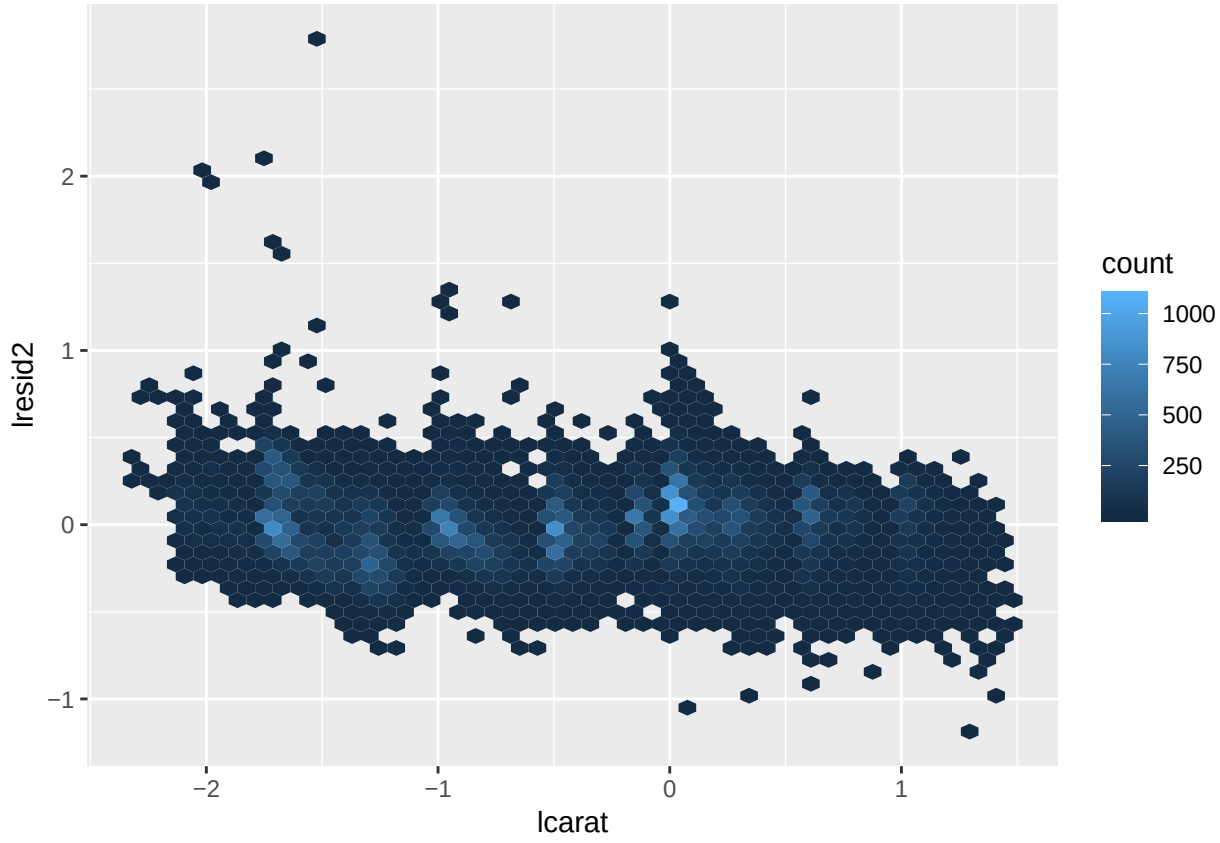
Let's include all 4 components into our linear model now.

```
mod_diamond2 <- lm(
  lprice ~ lcarat + color + cut + clarity,
  data = diamonds2
)
grid <- diamonds2 %>%
  # data_grid(cut, .model = mod_diamond2) %>%
  data_grid(cut, color, clarity,
            lcarat = seq_range(lcarat, 20)) %>%
  add_predictions(mod_diamond2)
grid
```

```
## # A tibble: 5,600 x 5
##    cut   color clarity lcarat  pred
##    <ord> <ord> <ord>    <dbl> <dbl>
##  1 Fair  D     I1      -2.32   6.98
##  2 Fair  D     I1      -2.12   7.36
##  3 Fair  D     I1      -1.92   7.74
##  4 Fair  D     I1      -1.72   8.12
##  5 Fair  D     I1      -1.52   8.49
##  6 Fair  D     I1      -1.32   8.87
##  7 Fair  D     I1      -1.12   9.25
##  8 Fair  D     I1      -0.919  9.63
##  9 Fair  D     I1      -0.719 10.0
## 10 Fair  D     I1      -0.518 10.4
```

```
## # ... with 5,590 more rows
```

```
diamonds2 <- diamonds2 %>%
  add_residuals(mod_diamond2, "lresid2")
ggplot(diamonds2, aes(lcarat, lresid2)) +
  geom_hex(bins = 50)
```




```
diamonds2 %>%
  filter(abs(lresid2) > 1.5) %>%
  add_predictions(mod_diamond2) %>%
  mutate(pred = round(2 ^ pred)) %>%
  select(price, pred, carat:table, x:z) %>%
  arrange(price)
```

```
## # A tibble: 7 x 11
##   price  pred carat cut       color clarity depth table     x     y     z
##   <int> <dbl> <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  1013   264 0.25  Fair      F     SI2      54.4    64  4.3   4.23  2.32
## 2  1186   285 0.25  Premium   G     SI2      59      60  5.33  5.28  3.12
## 3  1186   285 0.25  Premium   G     SI2      58.8    60  5.33  5.28  3.12
## 4  1715   577 0.32  Fair      F     VS2      59.6    60  4.42  4.34  2.61
## 5  1776   413 0.290 Fair      F     SI1      55.8    60  4.48  4.41  2.48
## 6  2160   313 0.34  Fair      F     I1       55.8    62  4.72  4.6   2.6
## 7  2366   775 0.3   Very Good D     VVS2     60.6    58  4.33  4.35  2.63
```

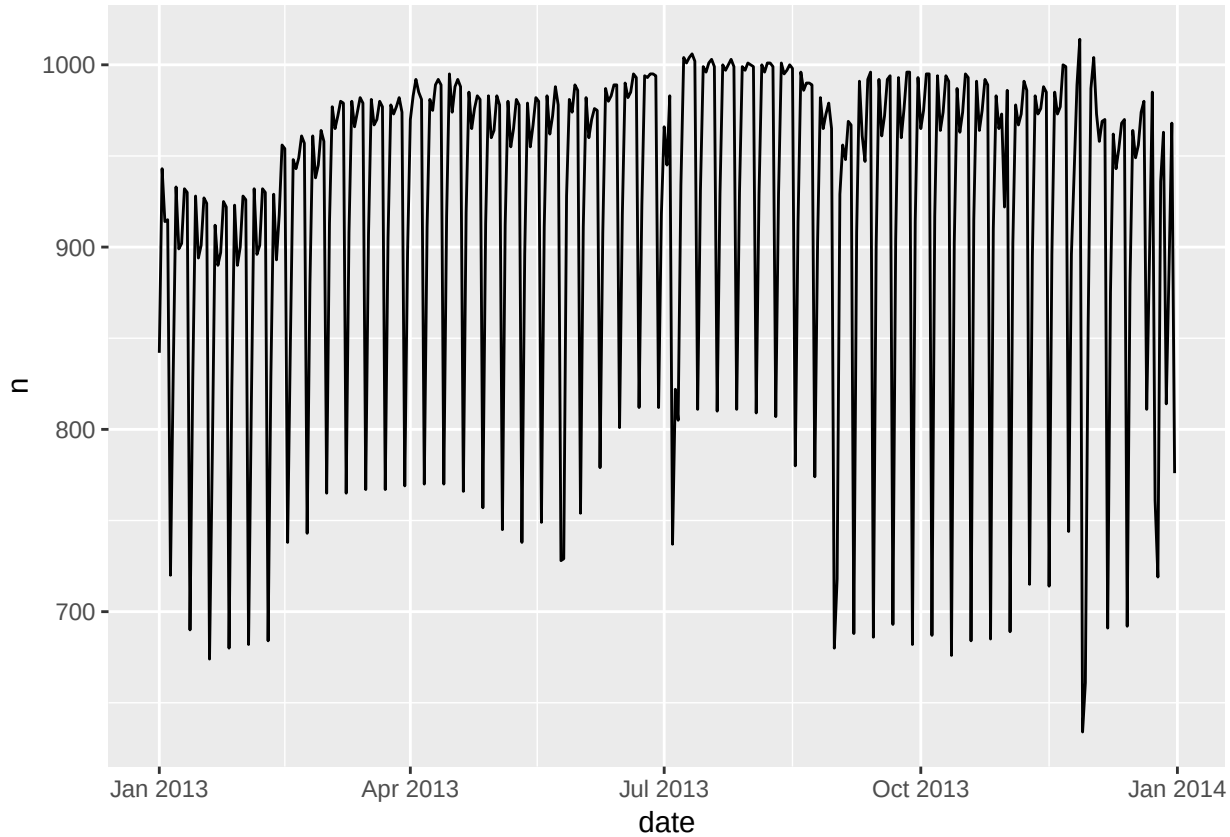
Number of Daily Flights

We will group the flights dataset by date and see if there is a pattern.

```
daily <- flights %>%
  mutate(date = make_date(year, month, day)) %>%
  count(date)
daily
```

```
## # A tibble: 365 x 2
##    date           n
##    <date>     <int>
##  1 2013-01-01   842
##  2 2013-01-02   943
##  3 2013-01-03   914
##  4 2013-01-04   915
##  5 2013-01-05   720
##  6 2013-01-06   832
##  7 2013-01-07   933
##  8 2013-01-08   899
##  9 2013-01-09   902
## 10 2013-01-10   932
## # ... with 355 more rows
```

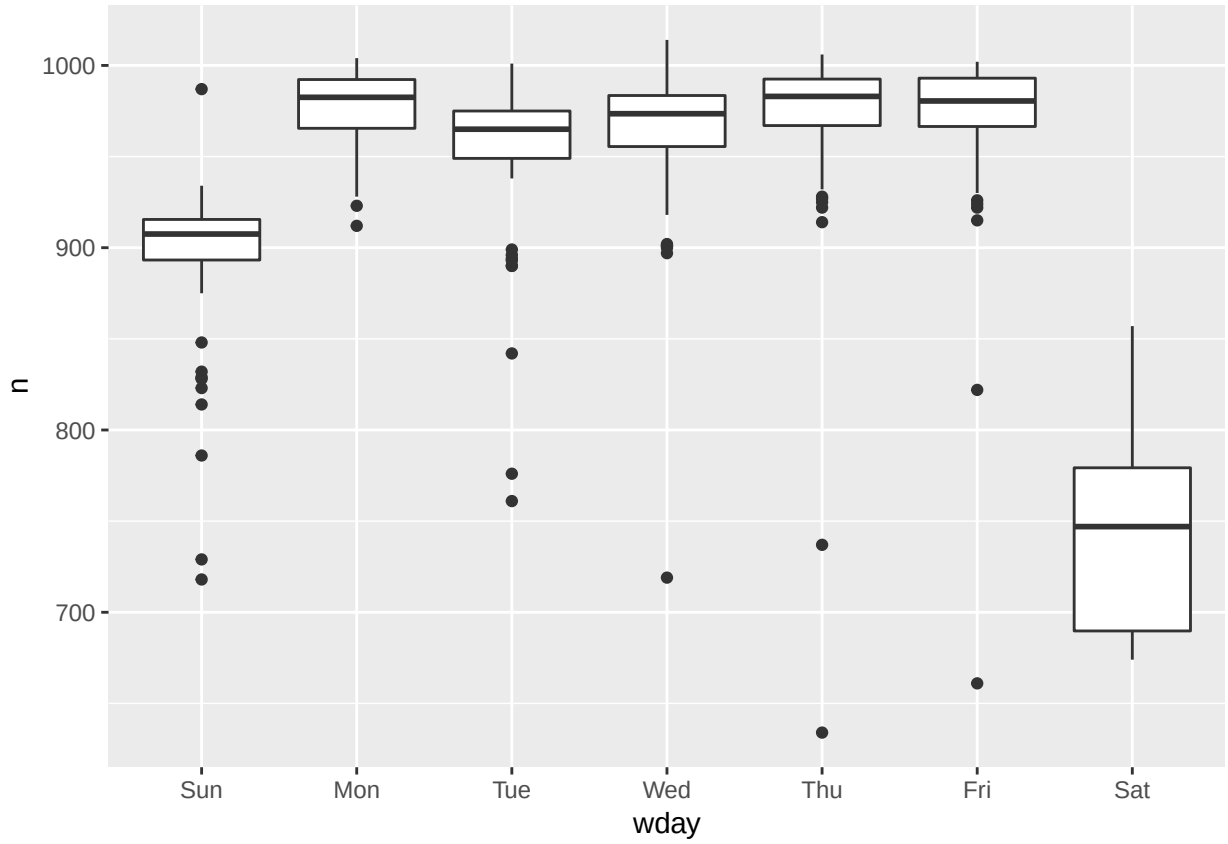
```
daily %>% ggplot(aes(date, n)) +
  geom_line()
```



It looks like the dips are regularly spaced. Perhaps it is day of the week? Let's see if we can figure out if that is at least part of the reason. We will use wday to see which day of the week it is and add that as a new

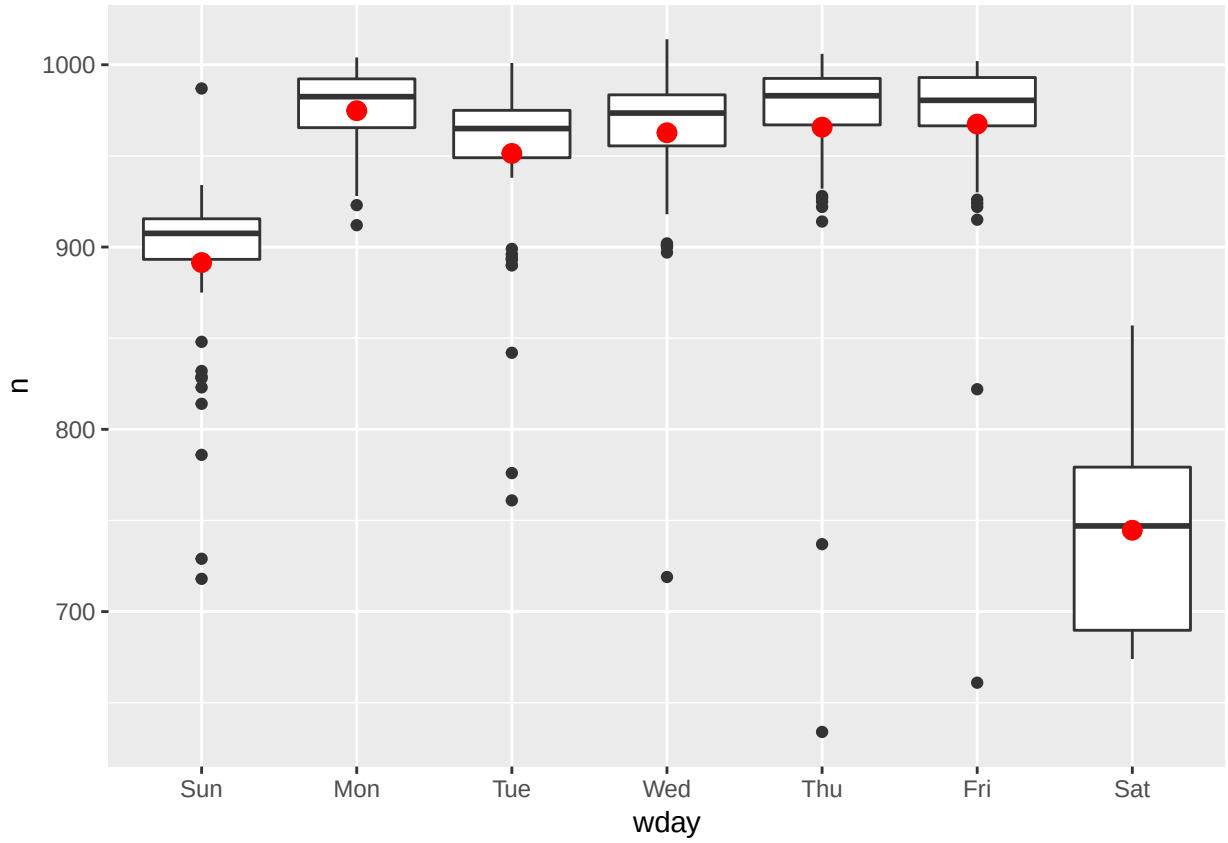
column to our dataset.

```
daily <- daily %>%
  mutate(wday = wday(date, label = TRUE))
g <- ggplot(daily, aes(wday, n)) +
  geom_boxplot()
g
```

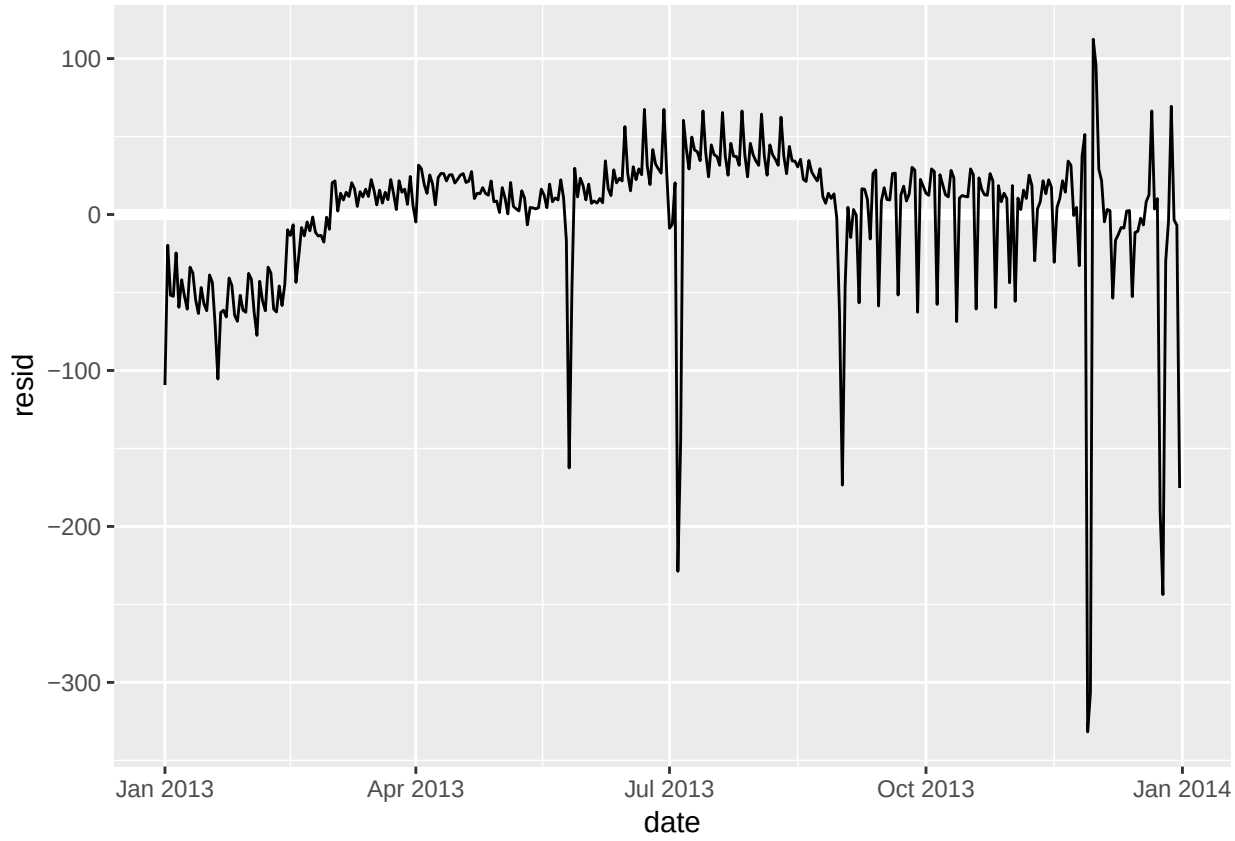


We will use a model to remove the strong pattern we see here, plotting the predictions on top of our boxplot chart, then looking at the residuals.

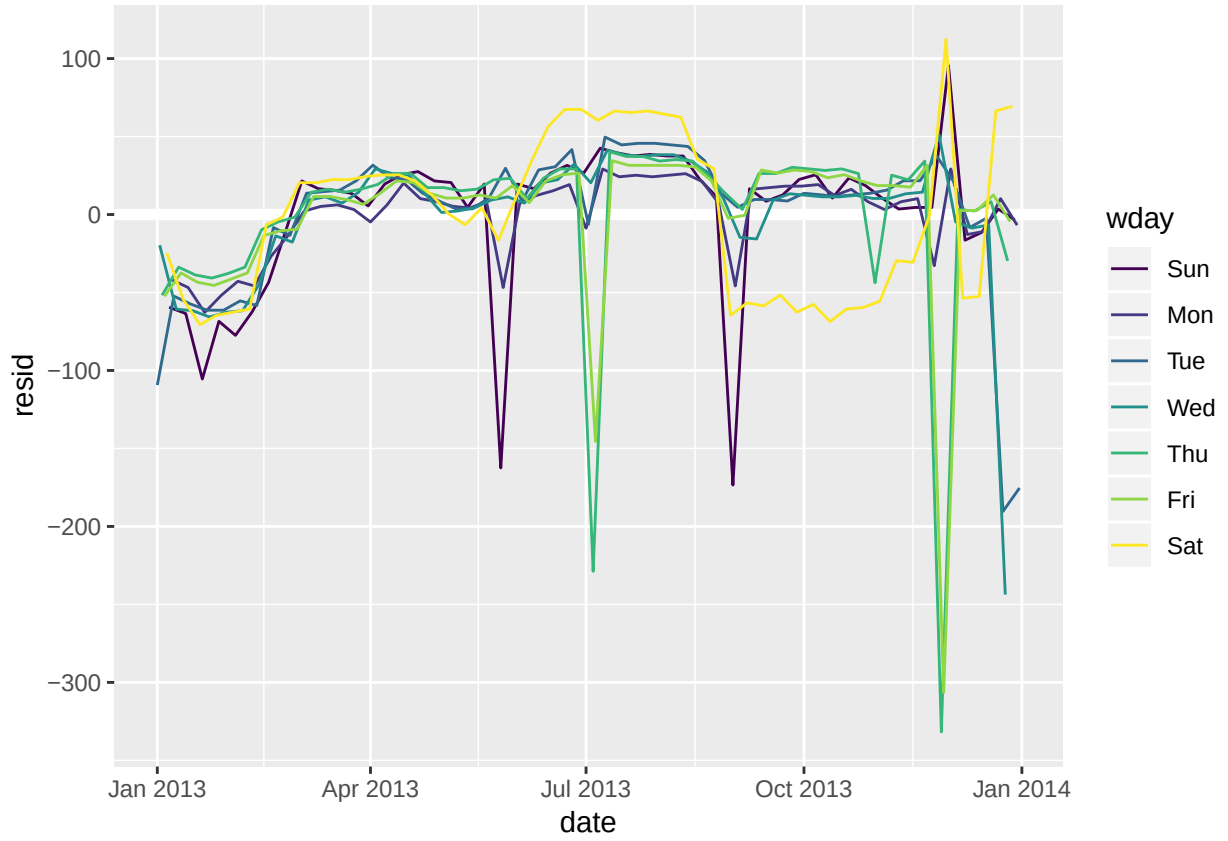
```
mod <- lm(n ~ wday, data = daily)

grid <- daily %>%
  data_grid(wday) %>%
  add_predictions(mod, "n")
g + geom_point(data = grid,
               color = "red", size = 3)
```




```
daily <- daily %>%
  add_residuals(mod)
daily %>%
  ggplot(aes(date, resid)) +
  geom_ref_line(h = 0) +
  geom_line()
```




```
# plot with lines for each weekday to see if that helps
daily %>%
  ggplot(aes(date, resid, color = wday)) +
  geom_ref_line(h = 0) +
  geom_line()
```



In particular,it looks like there may be some seasonal effect on the number of flights with a marked difference in the Saturday flights. We will look at those in a minute. First, let's look at those flight days with a large residual.

```
daily %>%
  filter(abs(resid) > 100 )
```

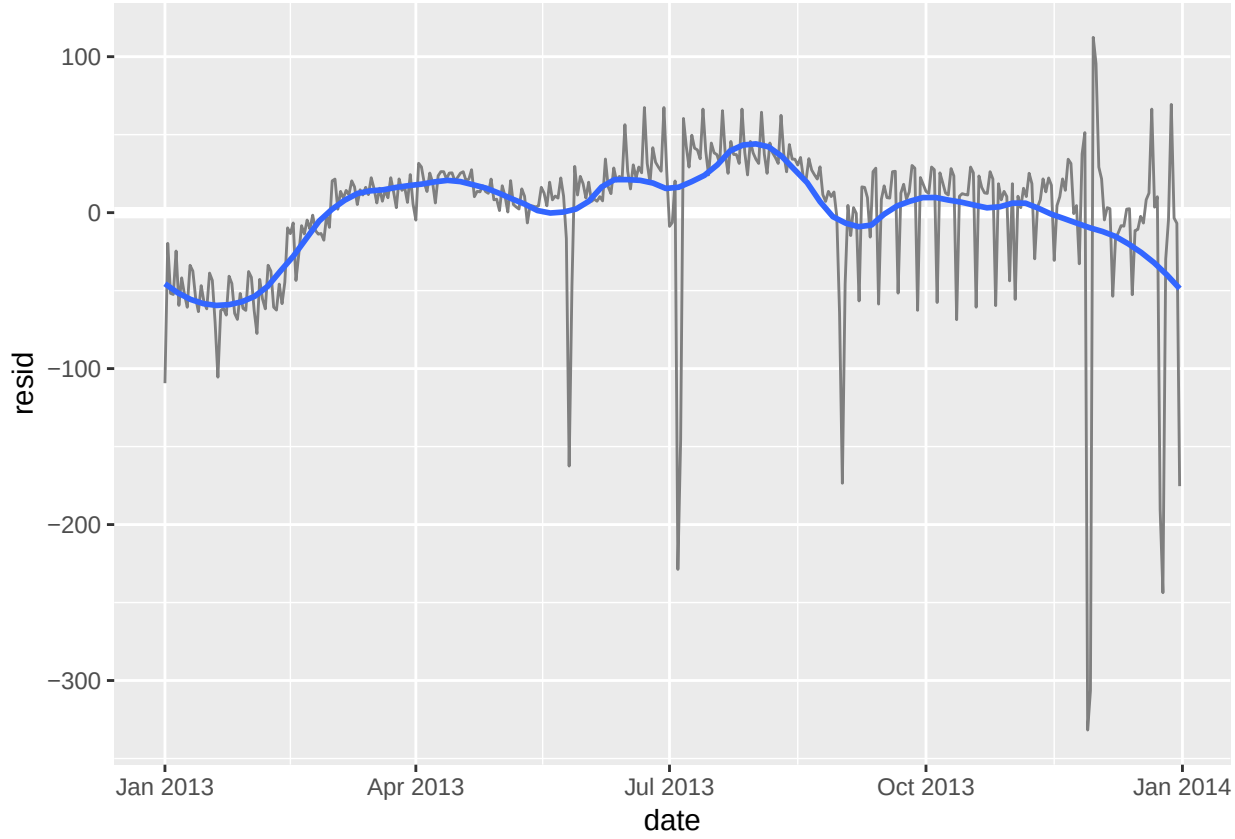
```
## # A tibble: 12 x 4
##    date           n wday  resid
##    <date>     <int> <ord> <dbl>
##  1 2013-01-01   842 Tue   -109.
##  2 2013-01-20   786 Sun   -105.
##  3 2013-05-26   729 Sun   -162.
##  4 2013-07-04   737 Thu   -229.
##  5 2013-07-05   822 Fri   -145.
##  6 2013-09-01   718 Sun   -173.
##  7 2013-11-28   634 Thu   -332.
##  8 2013-11-29   661 Fri   -306.
##  9 2013-11-30   857 Sat    112.
## 10 2013-12-24   761 Tue   -190.
## 11 2013-12-25   719 Wed   -244.
## 12 2013-12-31   776 Tue   -175.
```

What is the long-term trend over the course of a year?

```
daily %>%
  ggplot(aes(date, resid)) +
  geom_ref_line(h = 0) +
```

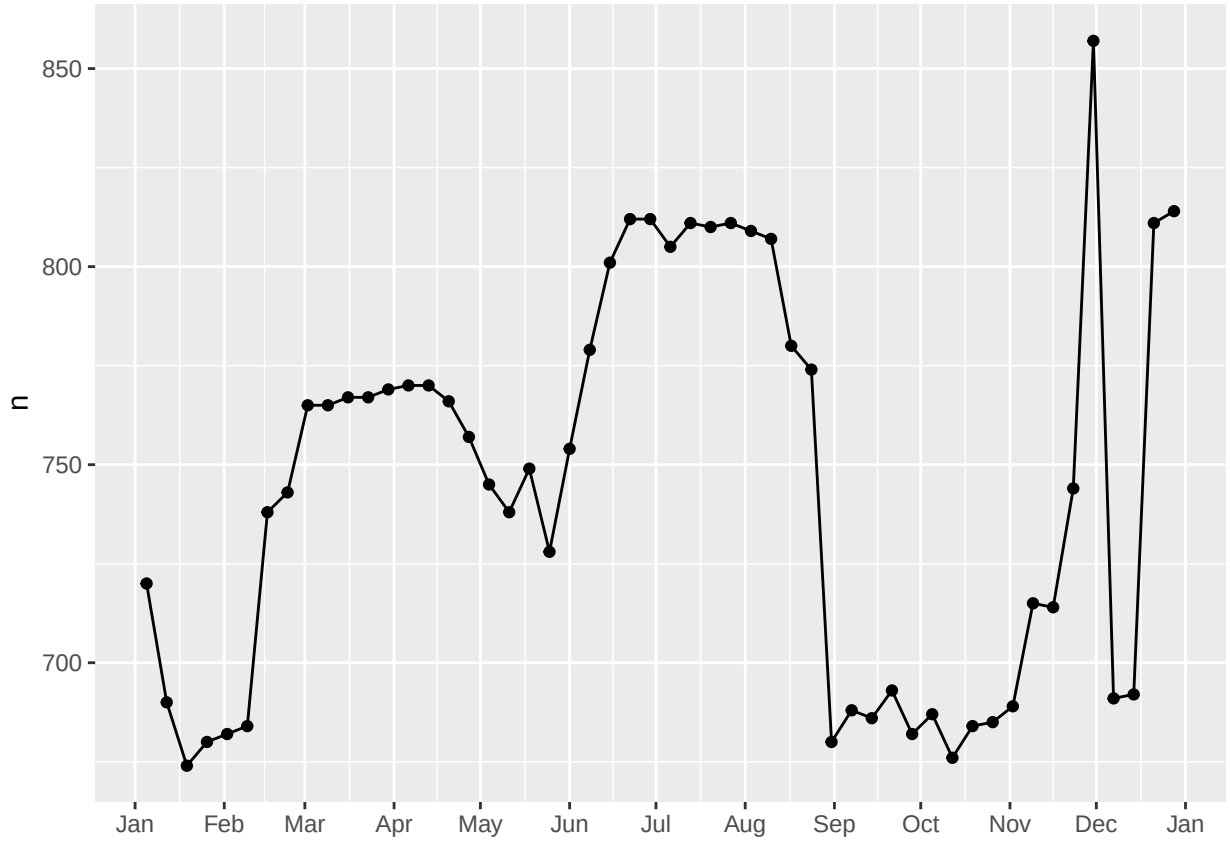
```
  geom_line(color = "grey50") +
  geom_smooth(se = FALSE, span = 0.20)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

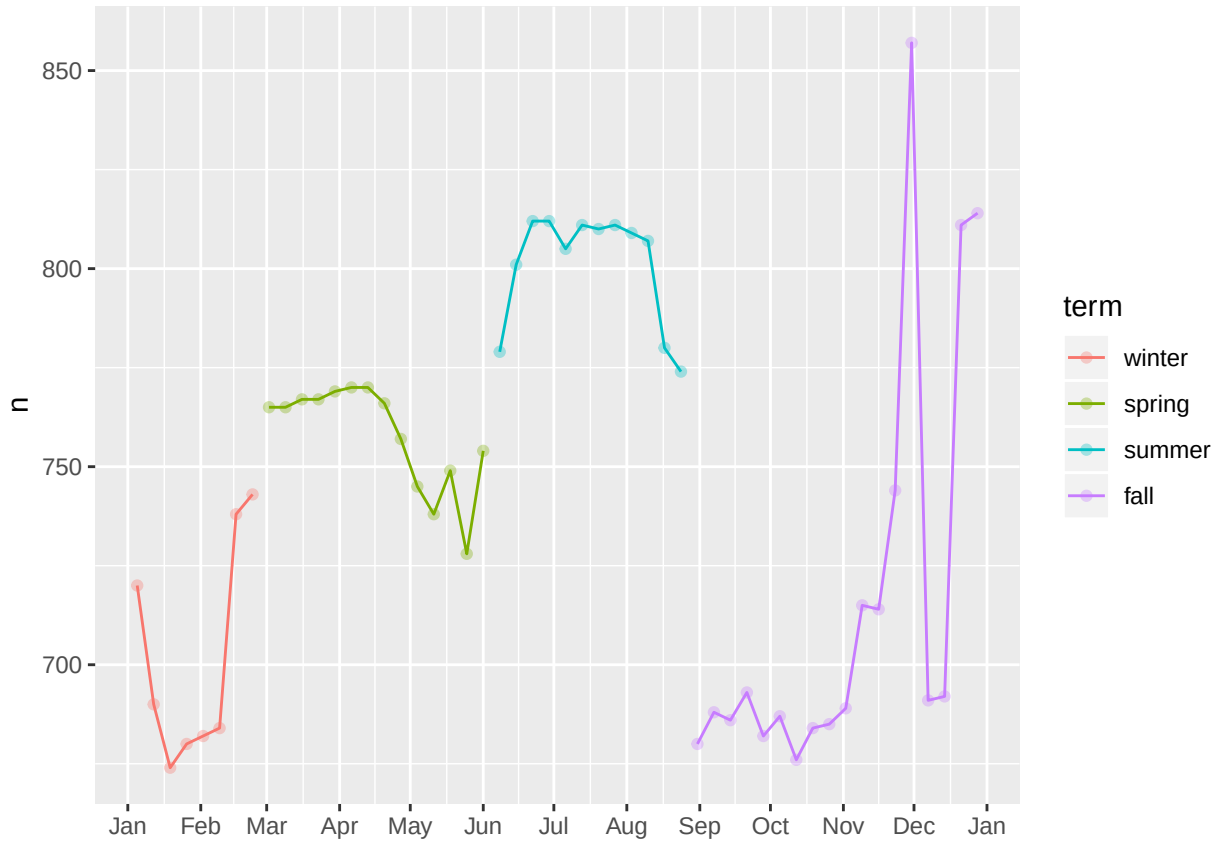


Let's look at just the Saturday data.

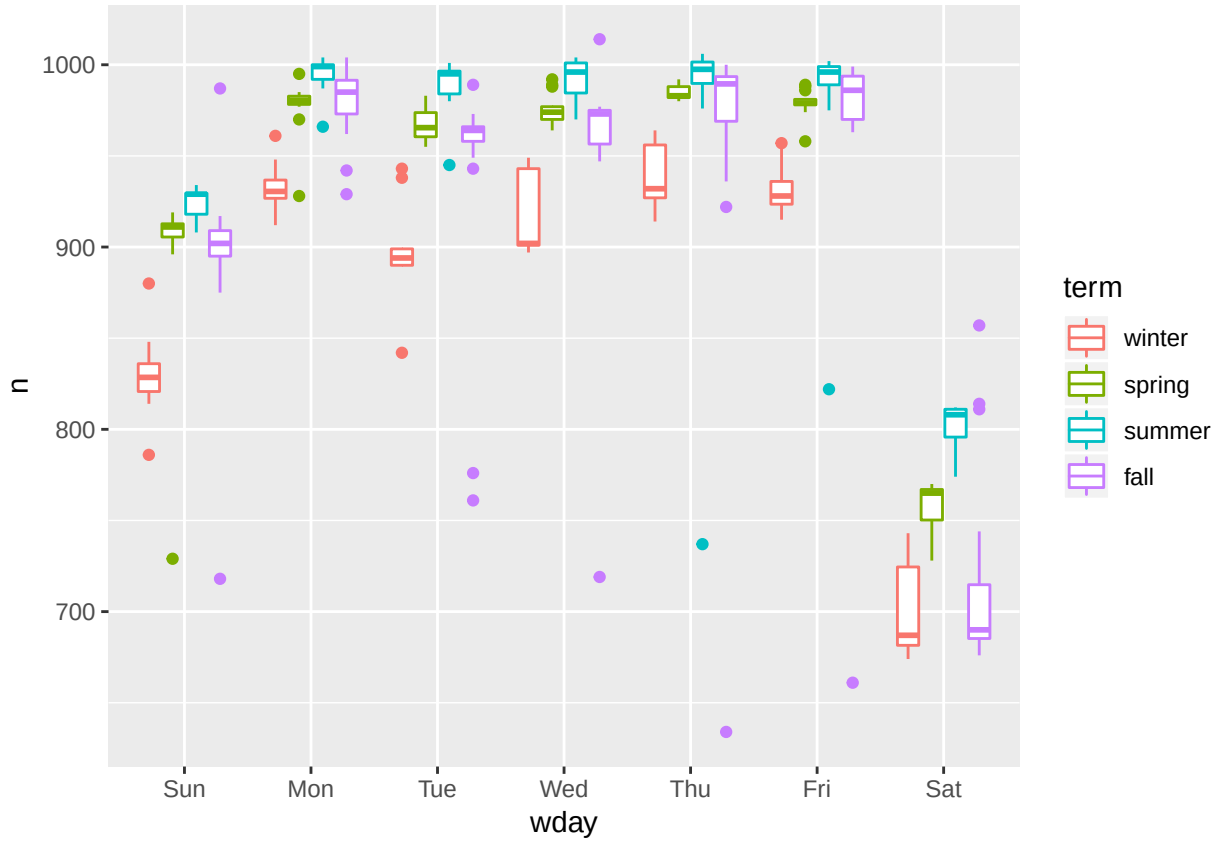
```
daily %>%
  filter(wday == "Sat") %>%
  ggplot(aes(date, n)) +
  geom_point() +
  geom_line() + scale_x_date(
    NULL,
    date_breaks = "1 month",
    date_labels = "%b"
  )
```




```
term <- function(date) {
  cut(date,
    breaks = ymd(20130101, 20130301, 20130605, 20130825, 20140101),
    labels = c("winter","spring", "summer", "fall")
  )
}

daily <- daily %>%
  mutate(term = term(date))

daily %>%
  filter(wday == "Sat") %>%
  ggplot(aes(date, n, colour = term)) +
  geom_point(alpha = 1/3) +
  geom_line() +
  scale_x_date(NULL, date_breaks = "1 month", date_labels = "%b")
```

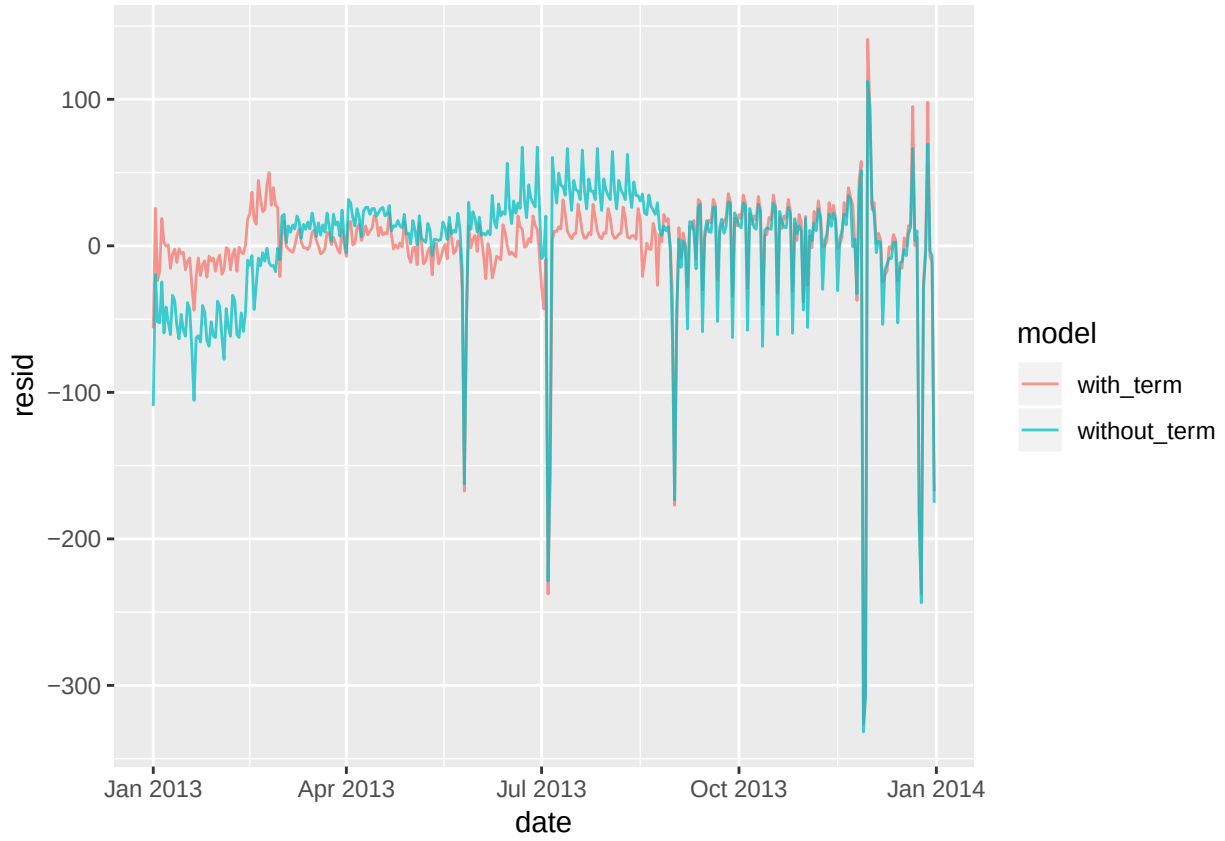



```
# look at how season effects other days
daily %>%
  ggplot(aes(wday, n, colour = term)) +
    geom_boxplot()
```

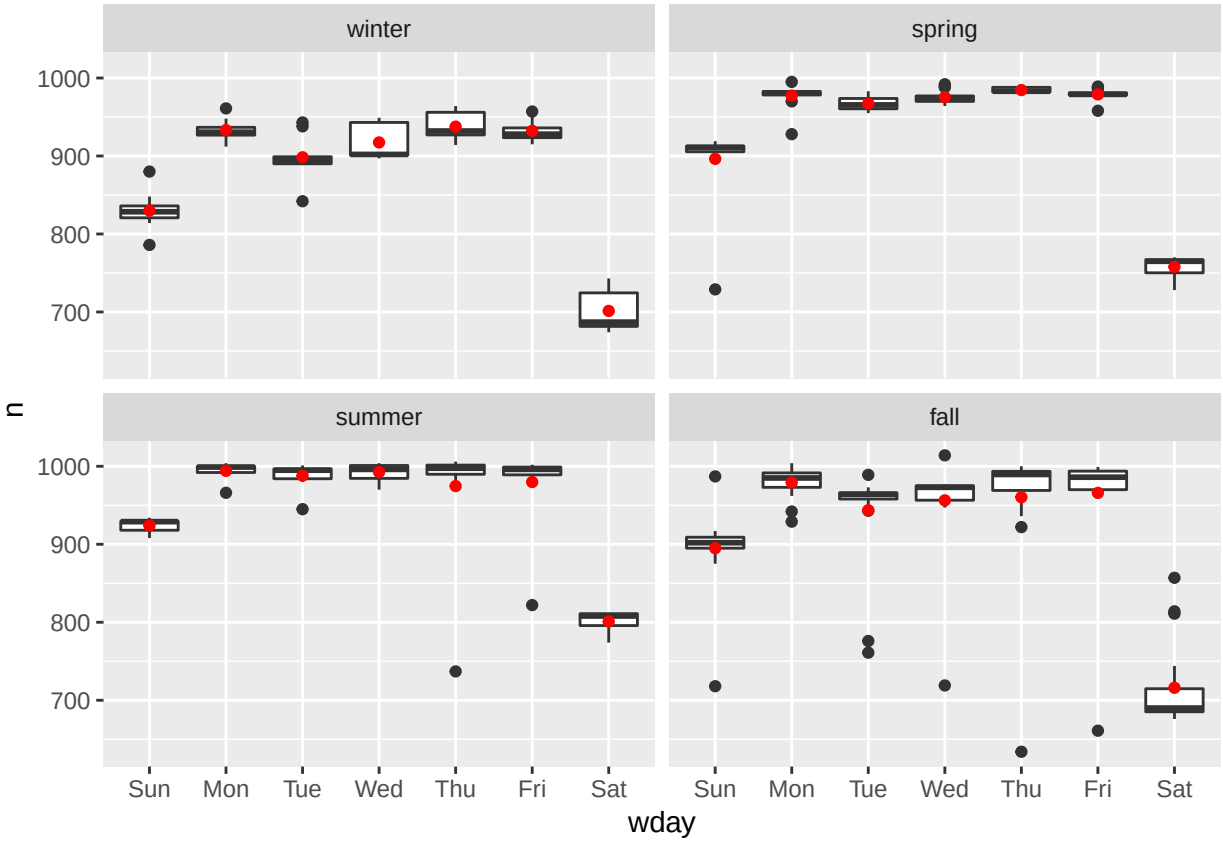


It looks like there is significant variation across the terms, so let's try adding the term into our model and see if that helps.

```
mod1 <- lm(n ~ wday, data = daily)
mod2 <- lm(n ~ wday * term, data = daily)

daily %>%
  gather_residuals(without_term = mod1, with_term = mod2) %>%
  ggplot(aes(date, resid, colour = model)) +
    geom_line(alpha = 0.75)
```

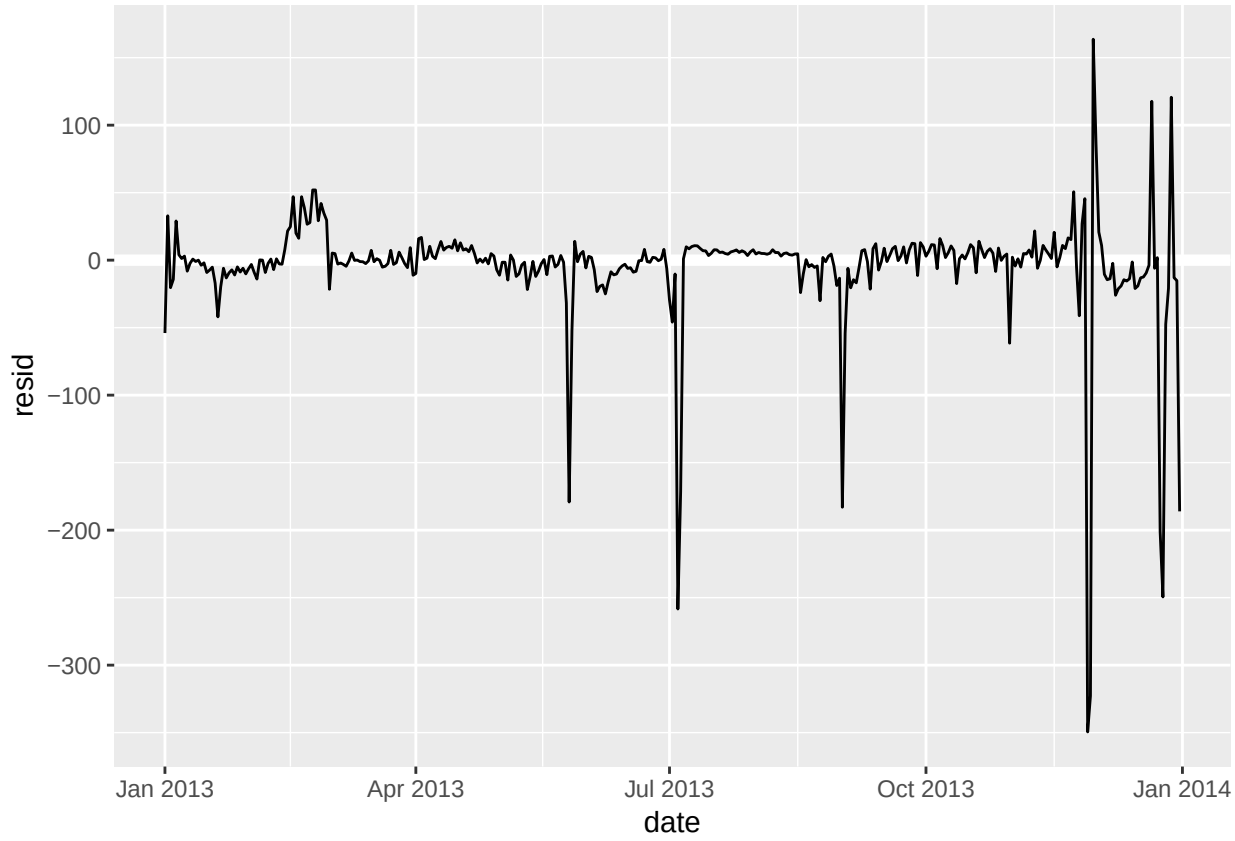



```
# overlay predicted values on the boxplots
grid <- daily %>%
  data_grid(wday, term) %>%
  add_predictions(mod2, "n")

ggplot(daily, aes(wday, n)) +
  geom_boxplot() +
  geom_point(data = grid, colour = "red") +
  facet_wrap(~ term)
```



The outliers are significantly altering our predictions since it finds the mean effect. Instead, use MASS::rlm() since it is more robust to outlier effects.

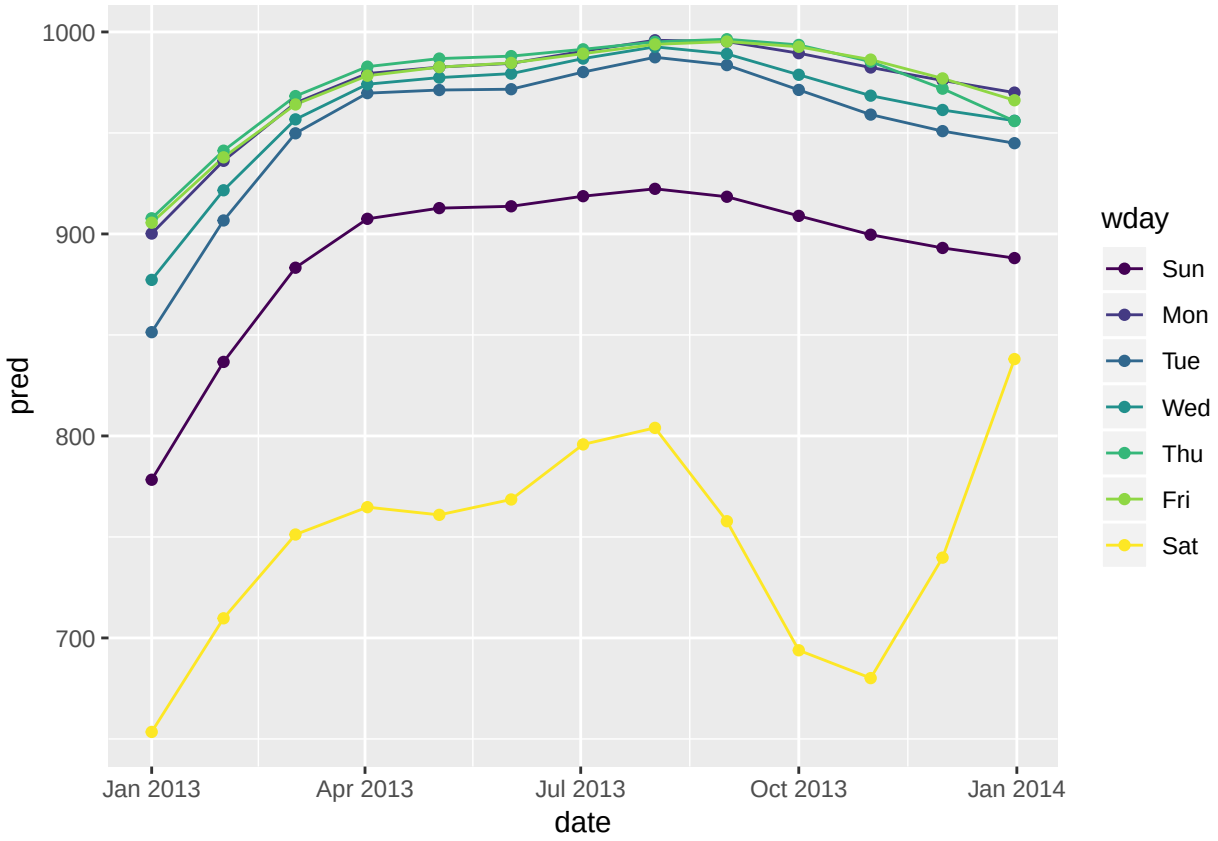
```
mod3 <- MASS::rlm(n ~ wday * term, data = daily)
daily %>%
  add_residuals(mod3, "resid") %>%
  ggplot(aes(date, resid)) +
  geom_hline(yintercept = 0, size = 2, colour = "white") +
  geom_line()
```



Time of Year: an alternate approach

We could instead just fit splines to the data broken out by day of the week.

```
library(splines)
mod <- MASS::rlm(n ~ wday * ns(date, 5), data = daily)

daily %>%
  data_grid(wday, date = seq_range(date, n = 13)) %>%
  add_predictions(mod) %>%
  ggplot(aes(date, pred, colour = wday)) +
    geom_line() +
    geom_point()
```



Homework: Given this data, what would you predict for 2019 and why?