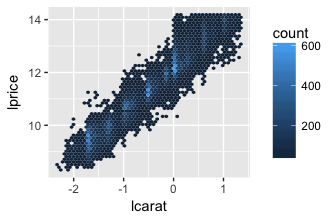
Final Exam

# 24.2.3 Exercises

## (1) 24.2.3 Exercise 1

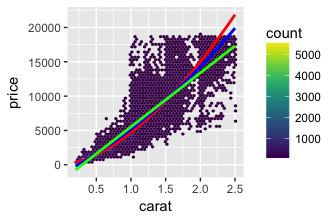
In the plot of lcarat vs. lprice, there are some bright vertical strips. What do they represent? Visually demonstrate your hypothesis with plot(s).

diamonds2 <- diamonds %>%   
 filter(carat <= 2.5) %>%   
 mutate(lprice = log2(price), lcarat = log2(carat))  
  
ggplot(diamonds2, aes(lcarat, lprice)) +   
 geom\_hex(bins = 50)



In a logarithmic scale a constant percentage trend is seen as a constant vertical or horizontal line, so these vertical concentrations of datum are percentage trends from the original dataset. I will attempt to show this below. Logarithmic scales will show exponential data trends as linear trends.

mod\_cp <- lm(lprice ~ lcarat, data=diamonds2)  
  
p\_grid <- diamonds2 %>%  
 data\_grid(carat = seq\_range(carat, 20)) %>%  
 mutate(lcarat = log2(carat)) %>%  
 add\_predictions(mod\_cp, "lprice") %>%  
 mutate(price = 2 ^ lprice)  
  
ggplot(diamonds2, aes(carat, price)) +  
 geom\_hex(bins = 50) +  
 scale\_fill\_viridis() +  
 geom\_line(data = p\_grid, color = "red", size = 1) +   
 geom\_smooth(method="lm", se=FALSE, color="blue", formula = y ~ poly(x,2)) +  
 geom\_smooth(color= "green", formula = y~x, method="lm", se=FALSE)



#diamonds %>% filter(carat <= 2.5) %>%  
 #mutate(scarat = carat^2) %>%  
 #ggplot(aes(carat, price)) +  
 #geom\_hex(bins = 100) +  
 #geom\_hex(bins = 100, color="red", x=carat, y=scarat) +  
#+ geom\_point(aes(x=carat^2, y=price, color="red"))  
 #geom\_smooth(method="lm", se=FALSE, color="red", formula = y ~ poly(x,2))  
   
 #diamonds %>% filter(carat <= 2.5) %>%  
 #ggplot(aes(carat, price)) +  
 #geom\_hex(bins = 100) +  
 #geom\_smooth(color= "green", formula = y~x, method="lm", se=FALSE)

We can see by plotting a quadratic regression line that it seems the data has an exponential trend much like we saw by the highlighted vertical lines in the logarithmic plot. We can see from the comparison of the two of these that there seems to be a stronger exponential trend than a linear one in the data. We can also see an exponential curve look from the graph above. The blue line represents the exponential curve. The red line represents the prediction after it has been transformed back into the original dataset versus the logarithmic plot. The green line represents the linear regression. We can see that there is a stronger exponential trend than linear. We could also look at the residuals for these graphs and see this trend. If we look at the residuals we would see how successful this prediction really is since it will remove the linear patter in the logarithmic data. This data above also tells us that large diamonds (bigger carat) are much cheaper than they should be because there is a cut off for price in this dataset.

## (2) 24.2.3 Exercise 2

If log(price) = a\_0 + a\_1 \* log(carat), what does that say about the relationship between price and carat?

That there is an exponential trend between the two that will result in a linear model if we map the logs. We can see that because the above appears very similar to a linear graph with the formula y=mx+b.

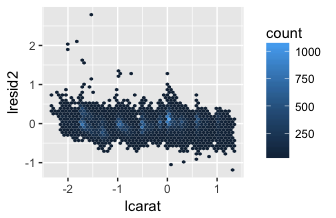
log(price) = a\_1 \* log(carat) + b

where log(price) = y, a\_1 = m (or slope), x = log(carat), and a\_0 = b (or y-intercept).

## (3) 24.2.3 Exercise 3

From the complicated model mod\_diamonds2 in the chapter, extract the diamonds that have very high and very low residuals. Is there anything unusual about these diamonds? Is there something else in the dataset that can explain their unexpected price, or do you think these are pricing errors?

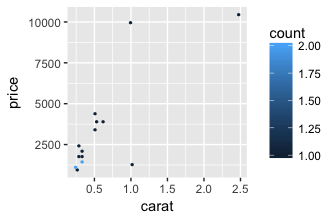
diamonds2 <- diamonds %>%   
 filter(carat <= 2.5) %>%   
 mutate(lprice = log2(price), lcarat = log2(carat))  
  
mod\_diamond <- lm(lprice ~ lcarat, data = diamonds2)  
  
diamonds2 <- diamonds2 %>%   
 add\_residuals(mod\_diamond, "lresid")  
  
mod\_diamond2 <- lm(lprice ~ lcarat + color + cut + clarity, data = diamonds2)  
  
diamonds2 <- diamonds2 %>%   
 add\_residuals(mod\_diamond2, "lresid2")  
  
ggplot(diamonds2, aes(lcarat, lresid2)) +   
 geom\_hex(bins = 50)



diamonds2a <- diamonds2 %>%   
 filter(abs(lresid2) > 1) %>%   
 add\_predictions(mod\_diamond2) %>%   
 mutate(pred = round(2 ^ pred)) %>%   
 select(price, pred, carat:table, x:z) %>%   
 arrange(price)  
diamonds2a

## # A tibble: 16 × 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 4.30 4.23 2.32  
## 2 1186 284 0.25 Premium G SI2 59.0 60 5.33 5.28 3.12  
## 3 1186 284 0.25 Premium G SI2 58.8 60 5.33 5.28 3.12  
## 4 1262 2644 1.03 Fair E I1 78.2 54 5.72 5.59 4.42  
## 5 1415 639 0.35 Fair G VS2 65.9 54 5.57 5.53 3.66  
## 6 1415 639 0.35 Fair G VS2 65.9 54 5.57 5.53 3.66  
## 7 1715 576 0.32 Fair F VS2 59.6 60 4.42 4.34 2.61  
## 8 1776 412 0.29 Fair F SI1 55.8 60 4.48 4.41 2.48  
## 9 2160 314 0.34 Fair F I1 55.8 62 4.72 4.60 2.60  
## 10 2366 774 0.30 Very Good D VVS2 60.6 58 4.33 4.35 2.63  
## 11 3360 1373 0.51 Premium F SI1 62.7 62 5.09 4.96 3.15  
## 12 3807 1540 0.61 Good F SI2 62.5 65 5.36 5.29 3.33  
## 13 3920 1705 0.51 Fair F VVS2 65.4 60 4.98 4.90 3.23  
## 14 4368 1705 0.51 Fair F VVS2 60.7 66 5.21 5.11 3.13  
## 15 10011 4048 1.01 Fair D SI2 64.6 58 6.25 6.20 4.02  
## 16 10470 23622 2.46 Premium E SI2 59.7 59 8.82 8.76 5.25

ggplot(diamonds2a, aes(carat, price)) +  
 geom\_hex(bins = 50)



We can see from this model that there are some diamonds that are very low in price in comparison with the rest of the diamonds as well as a couple that are very high in price in comparison with other diamonds of the same carat. These are the outliers with absolute value of 1 or greater than the other residuals making them a drastic price difference from the rest of the diamonds in the dataset. I looked at the rest of the parameters pertaining to these diamonds and they all seem reletavily normal/average. I believe these are most likely either pricing errors or possibly sales of some sort.

## (4) 24.2.3 Exercise 4

Does the final model, mod\_diamonds2, do a good job of predicting diamond prices? Would you trust it to tell you how much to spend if you were buying a diamond?

I would say that yes mod\_diamonds2 does a fairly good job of predicting diamond prices. I would trust the main portion of it and my analytical skills of the information to tell me how much to spend if I were buying a diamond. There are some outliers that we have seen, and that, as far as I can tell, do not have a reason for being such outliers. Overall, I would trust the mod\_diamond2 model in buying a diamond.

# 24.3.5 Exercises

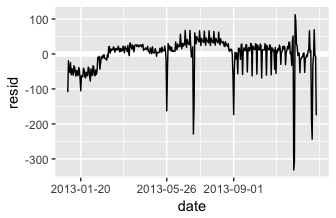
## (5) 24.3.5 Exercise 1

Come up with the likely reason why there were fewer than expected flights on Jan 20, May 26, and Sep 1. (Hint: they all have the same explanation.) How would these days generalize to another year?

daily <- flights %>%   
 mutate(date = make\_date(year, month, day)) %>%   
 group\_by(date) %>%   
 summarise(n = n())  
  
daily <- daily %>%   
 mutate(wday = wday(date, label = TRUE))  
  
mod <- lm(n ~ wday, data = daily)  
  
daily <- daily %>%   
 add\_residuals(mod)  
resid

## function (object, ...)   
## UseMethod("residuals")  
## <bytecode: 0x7fce4feff878>  
## <environment: namespace:stats>

ggplot(daily, aes(date, resid)) +  
 geom\_ref\_line(h = 0) +  
 geom\_line() +  
 scale\_x\_date(breaks = c(  
 ymd("2013-01-20"),  
 ymd("2013-05-26"),  
 ymd("2013-09-01")))



These days were all Sundays before federal holidays. January 20 was Inauguration Day and the next Day was Martin Luther King Jr. Day, a federal holiday. May 26 was the Sunday before Memorial Day. September 1 was the Sunday before labor day. These days generalized to another year may not have as much of an impact since these days of the month do not always fall on the same day of the week. These holidays are always on Mondays, so there will probably always be a drop in the number of flights the Sunday before. However, they will not always be on January 21, May 27, and September 2, and thus, these days will not always be the days of the month experiencing this dip.

## (6) 24.3.5 Exercise 2

What do the three days with high positive residuals represent? How would these days generalize to another year?

daily %>%  
 top\_n(3, resid)

## # A tibble: 3 × 4  
## date n wday resid  
## <date> <int> <ord> <dbl>  
## 1 2013-11-30 857 Sat 112.38462  
## 2 2013-12-01 987 Sun 95.51923  
## 3 2013-12-28 814 Sat 69.38462

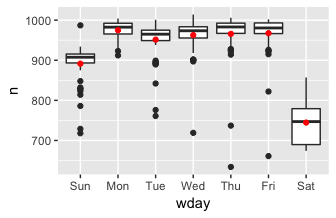
These represent the holidays of weekend after Thanksgiving travel and the weekend after Christmas travel. Christmas will translate to every year since it is always on the same day of the month and person will presumably always travel to see their families for this holiday. Similarly, Thanksgiving will most likely generalize to every year, but it will be a different day of the month each year.

## (7) 24.3.5 Exercises Exercise 3

Create a new variable that splits the wday variable into terms, but only for Saturdays; it should have Sun through Fri, but Sat-spring, Sat-summer, Sat-fall. How does this model compare with the model with every combination of wday and term?

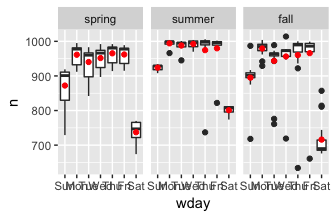
for each saturday examine term, then add the saturdays that are the same term into the same data variable

term <- function(date) {  
 cut(date,   
 breaks = ymd(20130101, 20130605, 20130825, 20140101),  
 labels = c("spring", "summer", "fall")   
 )  
}  
  
daily <- daily %>%   
 mutate(term = term(date))   
  
mod1 <- lm(n ~ wday, data = daily)  
  
grid <- daily %>%   
 data\_grid(wday) %>%   
 add\_predictions(mod1, "n")  
  
ggplot(daily, aes(wday, n)) +  
 geom\_boxplot() +   
 geom\_point(data = grid, colour = "red")



#map2(mu, sigma, rnorm, n = 5) %>% str()

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



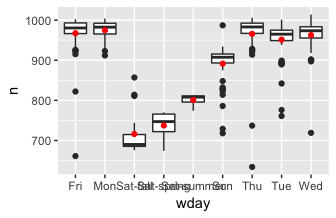
mod2 <- lm(n ~ wday, data = daily)  
  
extract\_term\_from\_Saturday <- function(data)  
{  
 data[, 3] <- sapply(data[, 3], as.character)  
 for(i in 1:nrow(data)){  
 day <- data[i, 3]  
 term <- data[i, 5]  
 if(day == "Sat")  
 {  
 if(term == 'spring')  
 {  
 data[i, 3] <- as.character("Sat-spring")  
 }  
 if(term == 'summer')  
 {  
 data[i, 3] <- as.character("Sat-summer")  
 }  
 if(term == 'fall')  
 {  
 data[i, 3] <- as.character("Sat-fall")  
 }  
 if(term == 'winter')  
 {  
 data[i, 3] <- "Sat-winter"  
 }  
 }  
 }  
 return(data)  
}  
daily3 <- extract\_term\_from\_Saturday(daily)  
daily3

## # A tibble: 365 × 5  
## date n wday resid term  
## <date> <int> <chr> <dbl> <fctr>  
## 1 2013-01-01 842 Tue -109.35849 spring  
## 2 2013-01-02 943 Wed -19.69231 spring  
## 3 2013-01-03 914 Thu -51.75000 spring  
## 4 2013-01-04 915 Fri -52.46154 spring  
## 5 2013-01-05 720 Sat-spring -24.61538 spring  
## 6 2013-01-06 832 Sun -59.48077 spring  
## 7 2013-01-07 933 Mon -41.80769 spring  
## 8 2013-01-08 899 Tue -52.35849 spring  
## 9 2013-01-09 902 Wed -60.69231 spring  
## 10 2013-01-10 932 Thu -33.75000 spring  
## # ... with 355 more rows

mod3 <- lm(n ~ wday, data = daily3)  
  
grid <- daily3 %>%   
 data\_grid(wday) %>%   
 add\_predictions(mod3, "n")  
grid

## # A tibble: 9 × 2  
## wday n  
## <chr> <dbl>  
## 1 Fri 967.4615  
## 2 Mon 974.8077  
## 3 Sat-fall 716.0000  
## 4 Sat-spring 737.3182  
## 5 Sat-summer 800.9167  
## 6 Sun 891.4808  
## 7 Thu 965.7500  
## 8 Tue 951.3585  
## 9 Wed 962.6923

daily3 %>%  
ggplot(aes(wday, n)) +  
 geom\_boxplot() +   
 geom\_point(data = grid, colour="red")

 Hints: Creating this variable is somewhat challenging, because it is a function of both wday and term, but term is only used when wday is “Sat”. Function such as mutate operate on whole vectors, but the logic here needs to operate on the components of each vector. In other words, daily is a table and wday and term are columns, but the logic needs to operate on each row. One way to do the calculation is to use a for loop, using an index such as “i” to access each row. However, I found the function map2\_chr(), described in 21.7, to be a more functional way of accomplishing the same thing. (Of course, there are probably other clever ways of calculating this variable.)

## (8) 24.3.5 Exercise 4

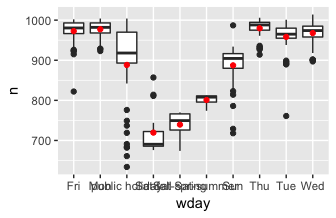
Create a new variable that combines wday, the day of week, term (for Saturdays), and public holidays. What do the residuals of that model look like? How does it compare to the model with every combination of wday and term?

I'm not exactly sure what is expected of me to accomplish here I can create a wday variable that combines day of the week and term for Saturdays as seen above. It will be extremely tough to do this for all public holidays unless I hardcode them into the function. I can definitely do that though.

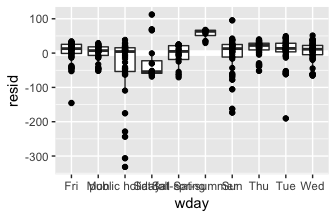
extract\_term\_from\_Saturday <- function(data)  
{  
 data[, 3] <- sapply(data[, 3], as.character)  
 for(i in 1:nrow(data)){  
 day <- data[i, 3]  
 term <- data[i, 5]  
 if(day == "Sat")  
 {  
 if(term == 'spring')  
 {  
 data[i, 3] <- as.character("Sat-spring")  
 }  
 if(term == 'summer')  
 {  
 data[i, 3] <- as.character("Sat-summer")  
 }  
 if(term == 'fall')  
 {  
 data[i, 3] <- as.character("Sat-fall")  
 }  
 if(term == 'winter')  
 {  
 data[i, 3] <- "Sat-winter"  
 }  
 }  
 }  
 return(data)  
}  
  
extract\_public\_holidays <- function(data)  
{  
 for(i in 1:nrow(data))  
 {  
 date <- data[i, 1]  
 if (date == '2013-01-01' | date == '2013-01-21' | date == '2013-01-24' |  
 date == '2013-02-02' | date == '2013-02-12' | date == '2013-02-14' | date == '2013-02-18' |  
 date == '2013-03-10' | date == '2013-03-17' | date == '2013-03-29' | date == '2013-03-31' |  
 date == '2013-04-01' | date == '2013-04-22' |  
 date == '2013-05-05' | date == '2013-05-12' | date == '2013-05-18' | date == '2013-05-19' | date == '2013-05-20' | date == '2013-05-27' |  
 date == '2013-06-14' | date == '2013-06-16' |  
 date == '2013-07-04' | date == '2013-07-28' |  
 date == '2013-09-02' | date == '2013-09-08' | date == '2013-09-11' | date == '2013-09-16' | date == '2013-09-17' | date == '2013-09-27' |   
 date == '2013-10-14' | date == '2013-10-16' | date == '2013-10-19' | date == '2013-10-31' |   
 date == '2013-11-03' | date == '2013-11-11' | date == '2013-11-28' | date == '2013-11-29' |   
 date == '2013-12-02' | date == '2013-12-07' | date == '2013-12-25' | date == '2013-12-31')  
 #holidays via https://www.calendar-365.com/holidays/2013.html  
 {  
 data[i, 3] <- as.character("public holiday")  
 }  
 }  
 return(data)  
}  
  
daily4 <- extract\_term\_from\_Saturday(daily)  
daily5 <- extract\_public\_holidays(daily4)  
  
mod5 <- lm(n ~ wday, data = daily5)  
  
grid <- daily5 %>%   
 data\_grid(wday) %>%   
 add\_predictions(mod5, "n")  
grid

## # A tibble: 10 × 2  
## wday n  
## <chr> <dbl>  
## 1 Fri 972.6667  
## 2 Mon 977.2927  
## 3 public holiday 888.3415  
## 4 Sat-fall 719.5625  
## 5 Sat-spring 739.5000  
## 6 Sat-summer 800.9167  
## 7 Sun 887.3333  
## 8 Thu 979.6809  
## 9 Tue 958.1633  
## 10 Wed 967.7551

daily5 %>%  
ggplot(aes(wday, n)) +  
 geom\_boxplot() +   
 geom\_point(data = grid, colour="red")



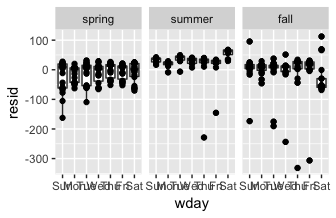
daily5 %>%  
 ggplot(aes(wday, resid)) +  
 geom\_ref\_line(h = 0) +   
 geom\_boxplot() +   
 geom\_point()



We can see from this that public holidays have several more outliers than do most days of the week. Sunday comes the closest after that. We can also tell from this boxplot that even the median of points is more spread out than regular days of the week.

We can compare this to every day and term by looking at the below.

mod2 <- lm(n ~ wday \* term, data = daily)  
  
grid <- daily %>%   
 data\_grid(wday, term) %>%   
 add\_predictions(mod2, "n")  
  
ggplot(daily, aes(wday, resid)) +  
 geom\_boxplot() +   
 geom\_point() +   
 facet\_wrap(~ term)



We can see from comparing these that some of the holidays that were the largest outliers in lowest number of flights occurred in the fall with a couple in the summer. We can tell this because these points match up between the two graphs.