In [1]:

```
library(tidyverse)
library(nycflights13)
```

```
── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──
✔ ggplot2 2.2.1     ✔ purrr   0.2.4
✔ tibble  1.4.1     ✔ dplyr   0.7.4
✔ tidyr   0.7.2     ✔ stringr 1.2.0
✔ readr   1.1.1     ✔ forcats 0.2.0
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

STATS 306

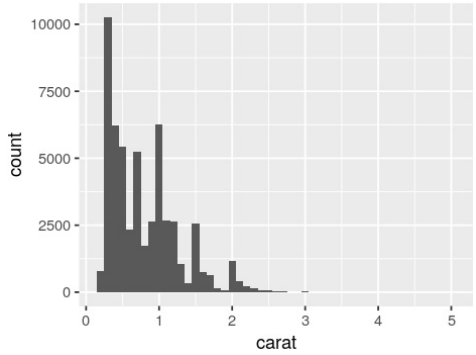
Problem set 4: exploratory data analysis

Each question is worth two points, for a total of 20.

Note: you do not need to use `install.packages()` in this notebook. You may assume that we have already installed all of the necessary packages when we run your code.

Problem 1

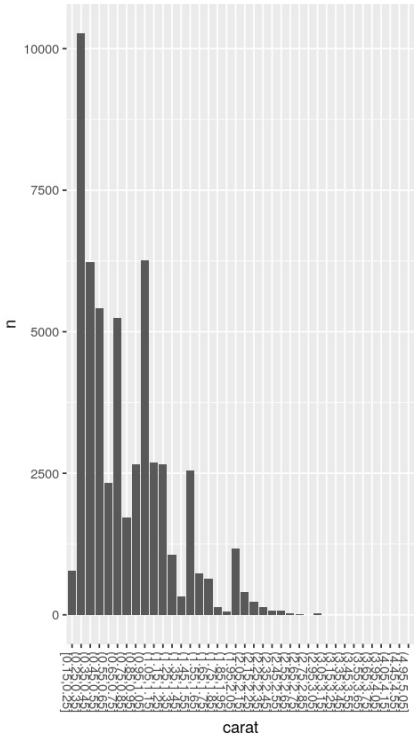
With a bin width of `.1` carats, the histogram for `diamonds$carat` looks like:



Use the functions `geom_col` and `cut_width` to re-create this plot. (It is okay if the bar spacing and x-axis appear differently in your plot.)

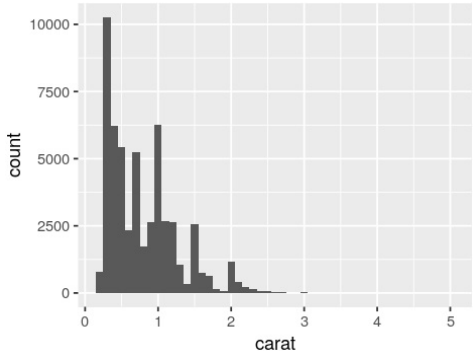
In [2]:

```
### BEGIN SOLUTION
data1 = mutate(diamonds, carat=cut_width(carat, .1)) %>% count(carat)
ggplot(data1) + geom_col(aes(x=carat, y=n)) +
                theme(axis.text.x = element_text(angle = -90, hjust = 1))
### END SOLUTION
```



Problem 2

Suppose you are given the following whisker plot for $n = 200$ samples of a random variable x:



1. About how many samples were between .1 and .2?
2. The upper whisker extends to about $x = 0.5$. Explain how this is calculated.

BEGIN SOLUTION

1. 25% of the data, or about 50 observations.
2. The $\text{IQR} = .2 - 0 = .2$ and the upper whisker is $q_{.75} + 1.5\text{IQR} = .2 + 1.5(.2) = .5$.

Code to generate plot:

```
{r}
options(jupyter.plot_mimetypes = "image/png")
set.seed(1)
x = rnorm(mean=.1, sd=.1/qnorm(.75), n=1000)
ggplot(data.frame(x=x)) + geom_boxplot(aes(x='', y=x)) + xlab('') + scale_y_continuous(breaks=(-3:7)/1
0)
print(x[x < -.3])
```

END SOLUTION

Problem 3

In the next few problems we will examine how departure delays covary between different air carriers in the NYC flights data set. First, determine the top six carriers in terms of the total number of flights they have in the data set. Store the result in `table3`. Your table should have two columns, `carrier` and `n` (the total number of flights in 2013), and six rows.

In [3]:

```
table3 = NA
### BEGIN SOLUTION
table3 = count(flights, carrier) %>% top_n(6, n)
### END SOLUTION
```

In [4]:

```
stopifnot(exists('table3'))
### BEGIN HIDDEN TESTS
table3_ans = count(flights, carrier) %>% top_n(6, n)
stopifnot(identical(
    arrange(table3, carrier),
    arrange(table3_ans, carrier)
))
### END HIDDEN TESTS
```

Problem 4

Next, drop all rows in `flights` that have missing `dep_delay` values, and then compute the daily median departure delay for the top six carriers that you found in problem 3.

- Use `table3`, the `filter()` function and `%in%` operator to programatically restrict to the top carriers in the `flights` table. Do not hardcode the carrier values.
- Store your result in `table4`.
- `table4` should have columns `year`, `month`, `day`, `carrier` and `med_dep_delay`.

In [5]:

```
table4 = NA
### BEGIN SOLUTION
table4 = filter(flights, !is.na(dep_delay), carrier %in% table3$carrier) %>%
                group_by(year, month, day, carrier) %>% summarize(med_dep_delay=median(dep_delay))
### END SOLUTION
```

In [6]:

```
stopifnot(exists("table4"))
### BEGIN HIDDEN TESTS
table4_ans = filter(flights, !is.na(dep_delay), carrier %in% table3$carrier) %>%
                group_by(year, month, day, carrier) %>%
                summarize(med_dep_delay=median(dep_delay))
stopifnot(identical(
    arrange(table4, year, month, day, carrier),
    arrange(table4_ans, year, month, day, carrier)))
### END HIDDEN TESTS
```

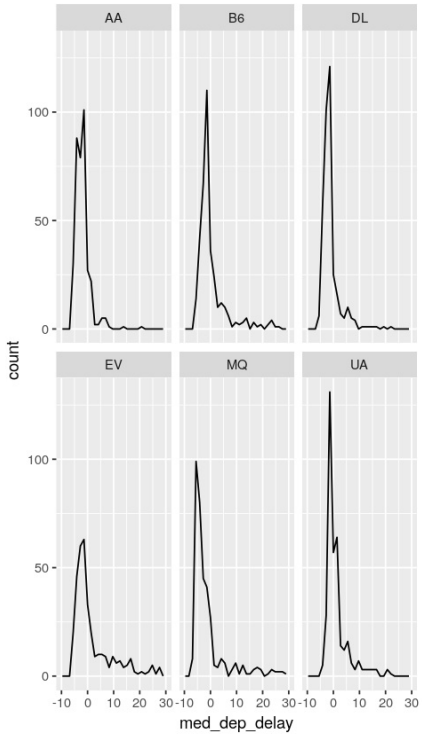
Problem 5

Use `facet_wrap` to plot a histogram of the median daily departure time for each air carrier in `table4`. Restrict the x-axis of each histogram to the range $[-10, 30]$ minutes.

In [7]:

```
### BEGIN SOLUTION
ggplot(table4) + geom_freqpoly(aes(x=med_dep_delay)) + facet_wrap(~ carrier) + xlim(c(-10, 30))
### END SOLUTION
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
Warning message:
“Removed 35 rows containing non-finite values (stat_bin).”Warning message:
“Removed 3 rows containing missing values (geom_path).”
```



Problem 6

Your data from table 4 should have one row per day per carrier. In order to study covariation between carriers, we need to *reshape* this data so that each carrier occupies a column, and there is one observation per day. We will study this sort of transformation soon. For now the command has been provided for you:

In [8]:

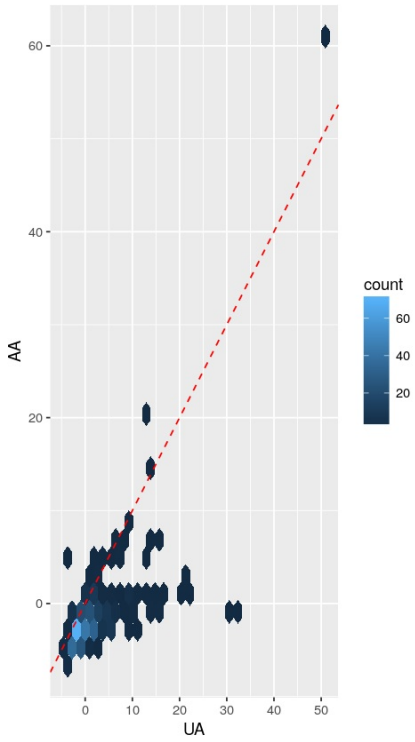
```
table4_wide = table4 %>% spread(key=carrier, value=med_dep_delay) %>% print
```

```
# A tibble: 365 x 9
# Groups:   year, month, day [365]
    year month   day    AA     B6    DL     EV    MQ    UA
 * <int> <int> <int> <dbl>  <dbl> <dbl>  <dbl> <dbl> <dbl>
 1  2013     1     1 -2.00  0.500 -3.50  10.0  -2.00  2.00
 2  2013     1     2  0    -1.00  -2.00  35.0  -2.50  3.00
 3  2013     1     3  0     1.50  -2.00   0    -4.00  2.00
 4  2013     1     4  0     3.00  -4.00   3.00 -5.00  1.00
 5  2013     1     5 -2.00  1.00  -3.00 - 2.50 -4.00  1.00
 6  2013     1     6 -3.00  4.00  -3.00   3.00 -4.00  2.00
 7  2013     1     7 -2.00 -2.00  -3.00 - 1.00 -5.00  1.00
 8  2013     1     8 -3.00 -1.00  -3.00 - 2.00 -6.00 -1.00
 9  2013     1     9 -4.00 -2.00  -4.00 - 5.00 -6.00 -2.00
10  2013     1    10 -4.00 -3.00  -4.00 - 4.00 -6.00 -2.00
# ... with 355 more rows
```

Use `table4_wide` to produce a hex plot of the median daily delay for American Airlines versus United Airlines. Also, use `geom_abline` to add a red, dashed 45-degree line. Which airline is more likely to have a departure delay according to the plot?

In [9]:

```
### BEGIN SOLUTION
ggplot(table4_wide) + geom_hex(aes(x=UA, y=AA)) + geom_abline(slope=1, linetype="dashed", color="red")
### END SOLUTION
```



Your answer here.

Football data

The next few questions use a new data set. The file `cfb.RData` contains a table called `cfb` with information on 5,116 college football games played since 2011. (More information about this data can be found here (http://www.seldomusedreserve.com/?page_id=8805).)

In [10]:

```
load('cfb.RData')
print(cfb)
```

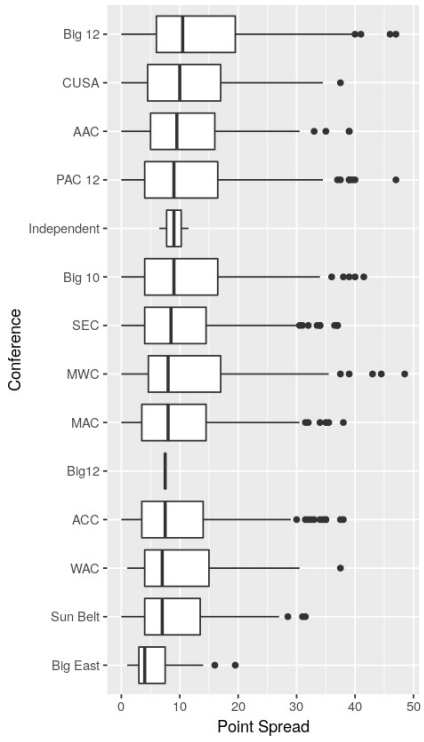
```
# A tibble: 5,116 x 50
    year  game count conf_… week  winni… winni… winni… winni… winn… winn… winn…
   <int> <int> <int> <chr>  <chr> <chr>  <chr>  <chr>   <int> <chr> <chr> <dbl>
 1  2011     1     1 Y      1     FIU    Sun B… H          41 F     Y     13.0
 2  2011     2     1 N      1     Wisco… Big 10 H          51 F     N     35.0
 3  2011     3     1 N      1     Missi… SEC    A          59 F     Y     30.5
 4  2011     4     1 N      1     Syrac… Big E… H          36 F     Y      6.00
 5  2011     5     1 N      1     Bowli… MAC    A          32 U     Y      6.00
 6  2011     6     1 N      1     Kentu… SEC    N          14 F     N     17.0
 7  2011     7     1 N      1     Baylor Big 12 H          50 U     Y      4.00
 8  2011     8     1 N      1     Ohio … Big 10 H          42 F     Y     31.5
 9  2011     9     1 N      1     Misso… Big 12 H          17 F     N     19.5
10  2011    10     1 N      1     Auburn SEC    H          42 F     N     24.0
# ... with 5,106 more rows, and 38 more variables: winning_o_u <chr>,
#   winning_passes <int>, winning_pass_yards <int>, winning_yppa <dbl>,
#   winning_rushes <int>, winning_rush_yards <int>, winning_ypra <dbl>,
#   winning_plays <int>, winning_total_yards <int>, winning_ypp <dbl>,
#   winning_to <int>, winning_pen_yards <int>, winning_top <int>,
#   field_25 <chr>, losing <chr>, losing_conference <chr>, losing_h_a_n <chr>,
#   losing_points <int>, losing_f_u <chr>, losing_cover <chr>,
#   losing_spread <dbl>, losing_o_u <chr>, losing_passes <int>,
#   losing_pass_yards <int>, losing_yppa <dbl>, losing_rush_attempts <int>,
#   losing_rush_yards <int>, losing_ypra <dbl>, losing_total_plays <int>,
#   losing_total_yards <int>, losing_ypp <dbl>, losing_to <int>,
#   losing_pen_yards <int>, losing_top <int>, `ot?` <chr>, `thursday?` <chr>,
#   top_verification <int>, o_u_total <chr>
```

Problem 7

Certain conferences have a reputation for being higher scoring. Investigate this by restricting to conference games and producing a box-and-whisker plot that shows the distribution of the winning point spreads by conference. The plot should be rotated 90 degrees so that the conference names are legible, and the conferences should be in descending order of the median point spread (i.e. conference with the highest median point spread is at the top of the plot.)

In [11]:

```
### BEGIN SOLUTION
filter(cfb, conf_game == "Y", !is.na(winning_spread)) %>% ggplot +
    geom_boxplot(aes(x=reorder(winning_conference, winning_spread, FUN=median),
                     y=winning_spread)) + xlab("Conference") + ylab("Point Spread") +
    coord_flip()
### END SOLUTION
```



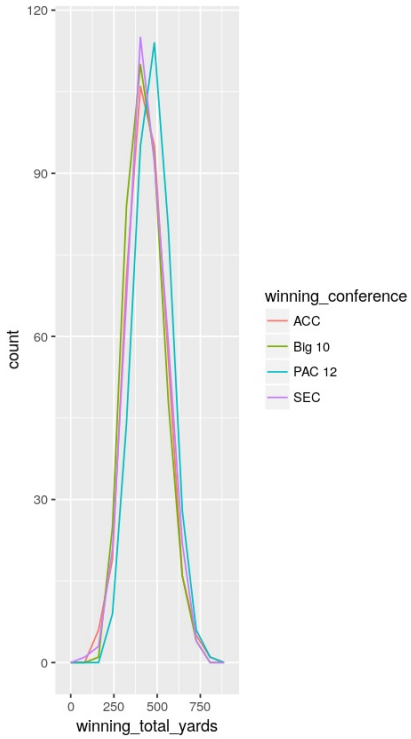
Problem 8

Filter `cfb` by again restricting to conference games, and then to the four largest conferences in terms of total games played. For each conference, plot a frequency polygon of the winning team's total yardage.

- Each conference should have a different colored line.
- To facilitate comparison between the different conferences, each frequency polygon should be normalized to have unit area.
- Experiment with a few values of the bin size to find a setting that provides an acceptable tradeoff between smoothness and detail. (There is no single correct answer for this part.)

In [12]:

```
### BEGIN SOLUTION
top_confs = filter(cfb, conf_game=="Y") %>% group_by(winning_conference) %>% count %>%
            ungroup %>%  top_n(4, n)
filter(cfb, conf_game=="Y", winning_conference %in% top_confs$winning_conference) %>%
    ggplot + geom_freqpoly(aes(x=winning_total_yards, color=winning_conference), bins=10)
### END SOLUTION
```

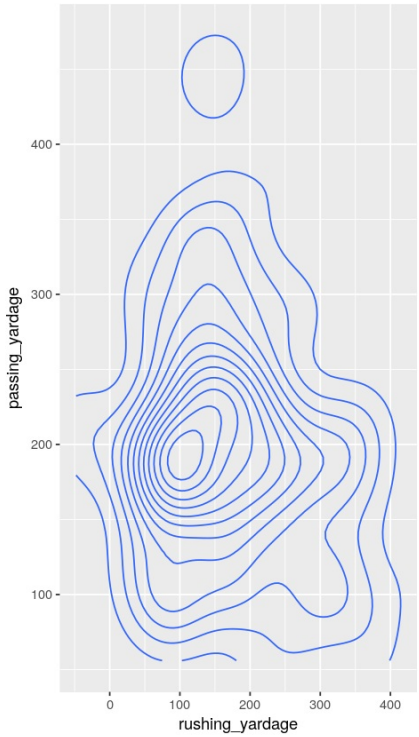


Problem 9

Compute a 2D density estimate (contour plot) of the joint distribution of Michigan's passing and rushing yardage across every University of Michigan game in `cfb`. Give your plot appropirate axis labels and a title.

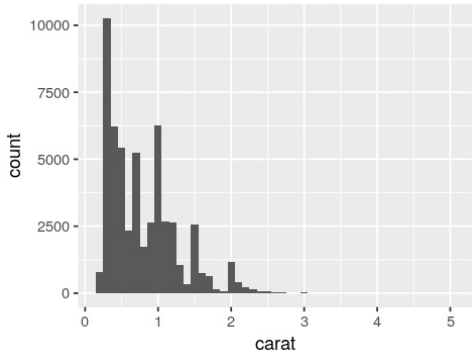
In [13]:

```
### BEGIN SOLUTION
umich = filter(cfb, winning == "Michigan" | losing == "Michigan") %>%
    mutate(
        passing_yardage=ifelse(
            winning=="Michigan",
            winning_pass_yards,
            losing_pass_yards
    ),
        rushing_yardage=ifelse(
            winning=="Michigan",
            winning_rush_yards,
            losing_rush_yards)
    )
ggplot(umich) + geom_density_2d(aes(x=rushing_yardage, y=passing_yardage))
### END SOLUTION
```



Problem 10

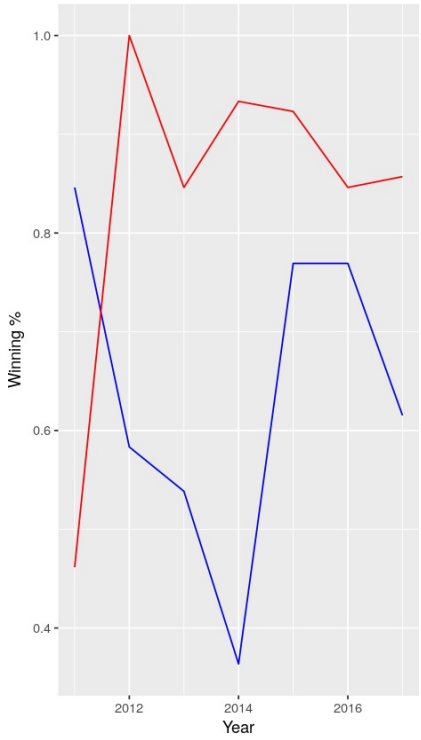
Compute the yearly winning percentage for Michigan and Ohio State and use it to generate the following line plot:



The plot should have a blue line for Michigan and a red line for Ohio, and as many x values as there are years in the data. (Note: I am not a football fan, but have been told this sort of graphic causes Michigan fans deep personal anguish. Feel free to annotate your plot accordingly if so.)

In [14]:

```
### BEGIN SOLUTION
umich_ohio = cfb %>%
             mutate(mich_game = winning == "Michigan" | losing == "Michigan",
                    oh_game = winning == "Ohio State" | losing == "Ohio State") %>%
             filter(mich_game | oh_game) %>%
             mutate(mich_win = ifelse(mich_game, winning == "Michigan", NA),
                    oh_win = ifelse(oh_game, winning == "Ohio State", NA)) %>%
             group_by(year) %>%
             summarize(mich_wpct = mean(mich_win, na.rm=T),
                       oh_wpct = mean(oh_win, na.rm=T))
ggplot(umich_ohio) + geom_line(aes(x=year, y=mich_wpct), colour="blue") +
                     geom_line(aes(x=year, y=oh_wpct), colour="red") + ylab("Winning %") + xlab("Year")
### END SOLUTION
```



In [15]:

```
umich_ohio
```

```
  year mich_wpct oh_wpct
1 2011 0.8461538 0.4615385
2 2012 0.5833333 1.0000000
3 2013 0.5384615 0.8461538
4 2014 0.3636364 0.9333333
5 2015 0.7692308 0.9230769
6 2016 0.7692308 0.8461538
7 2017 0.6153846 0.8571429
```