Lab_02R

36-290 – Statistical Research Methodology

Week 2 Thursday – Fall 2021

Data

We'll importing the same data that we used for Tuesday's lab, with rows with data with value 99 removed:

```
suppressMessages(library(tidyverse))
rm(list=ls())
file.path = "https://raw.githubusercontent.com/pefreeman/36-290/master/EXAMPLE_DATASETS/BUZZARD/Buzzard_DC1.Rdata"
load(url(file.path))
rm(file.path)
set.seed(101)
s = sample(nrow(df),4000)
df = df[s,]
df %>% filter(.,u!=99&g!=99&r!=99&i!=99&z!=99&y!=99) -> df.new
predictors = df.new[,-c(7:14)]
response   = as.vector(df.new[,14])

type = rep("FAINT",nrow(predictors))
w = which(predictors$i<25)
type[w] = "BRIGHT"
type = factor(type)
predictors = cbind(type,predictors)

rm(df,df.new,s,type,w)
objects()
```

```
## [1] "predictors" "response"
```

If everything loaded correctly, you should see two variables in your global environment: `predictors` and `response`. `predictors` is a data frame with 3624 rows and 6 columns, and `response` is a vector of length 3624, and it represents redshift.

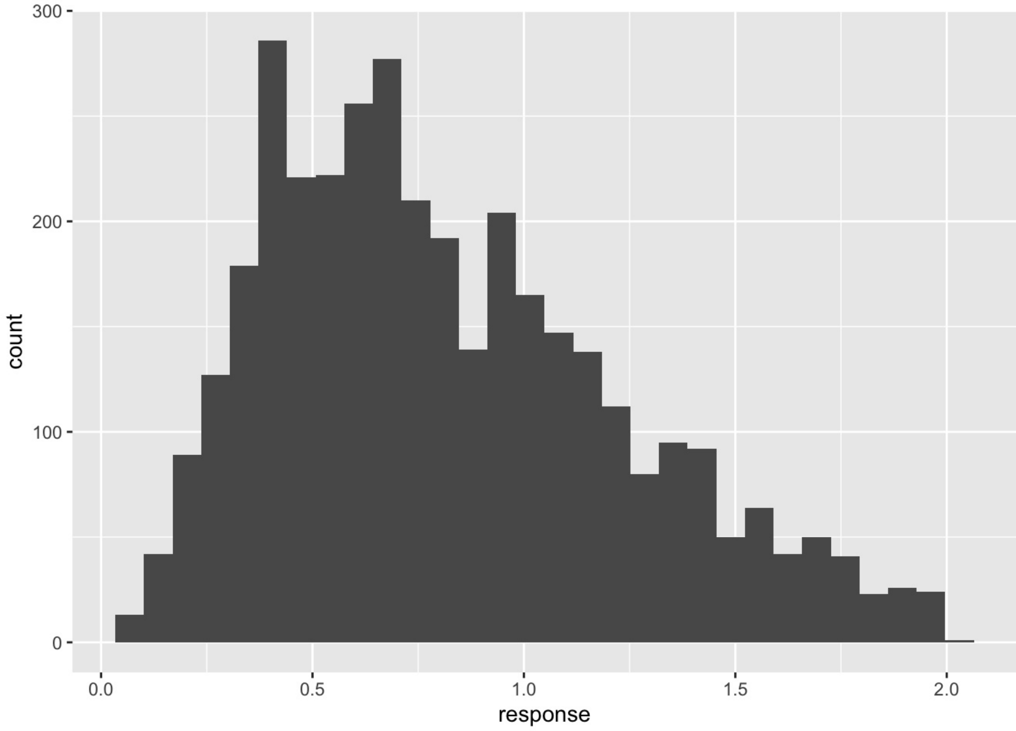
ggplot

In this course, we will use `ggplot` to create our plots. You will learn much about `ggplot` in 36-315; here, we will use it relatively simply, to make histograms and boxplots and scatter plots and the like. In order to become comfortable quickly with `ggplot`, you should read through Chapter 3 (online; Chapter 1 in print) of *R for Data Science* by Wickham and Grolemund, available for free here (http://r4ds.had.co.nz/).

The following is an example of a `ggplot` call:

```
ggplot(data=data.frame(response),mapping=aes(x=response)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



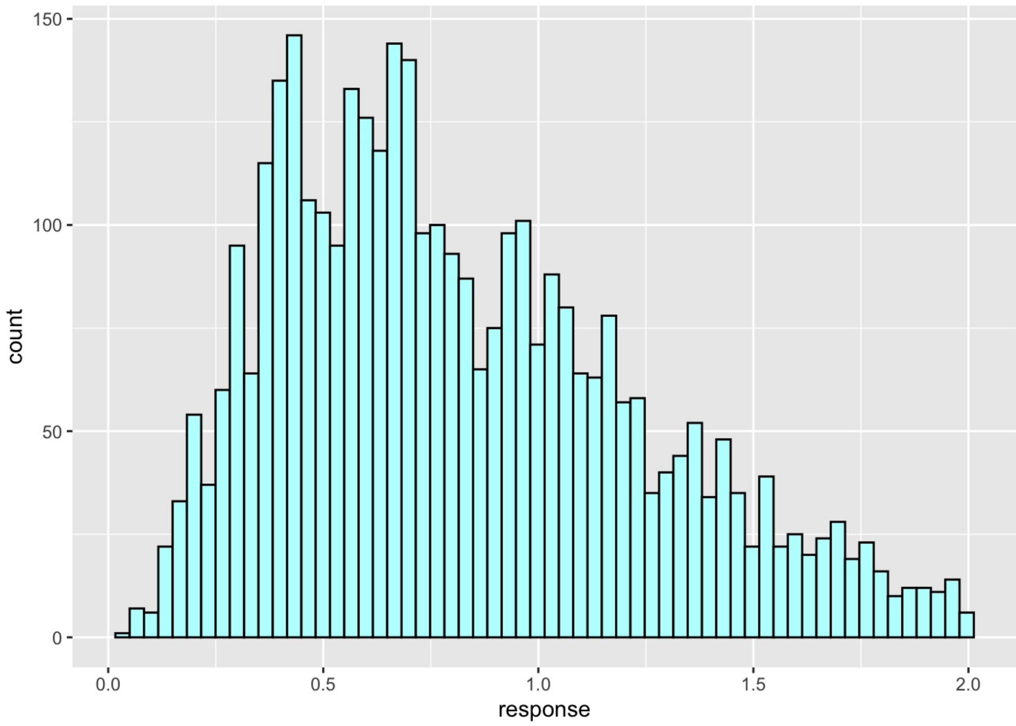
Note that `ggplot` expects data frames (or tibbles…), so I had to add the call to `data.frame`. Also note the structure here: the first function is `ggplot()`, and you pass a data frame (as `data`) and an identification of which feature goes along which axis (as `mapping`). Once you've identified the data and the mapping, you "add on" another function that describes what you will do to the data; in this case, that function is `geom_histogram()`.

Questions

Question 1

Re-run ggplot from above, doubling the number of bins. (Use the documentation for `geom_histogram()` to figure out how to do this…e.g., type `?geom_histogram` in the Console pane.) Also, change the color of the histogram via the `fill` argument. Google "R colors" to find listings of the names of `R`'s colors. My personal favorite is "papayawhip". Add a comment within the code chunk describing the empirical distribution of redshifts (where's the mode? is it bimodal? is it skew? why might it have the properties it does? etc.)

```
ggplot(data=data.frame(response),mapping=aes(x=response)) + geom_histogram(color="black", fill="paleturquoise1",bin
s=60)
```

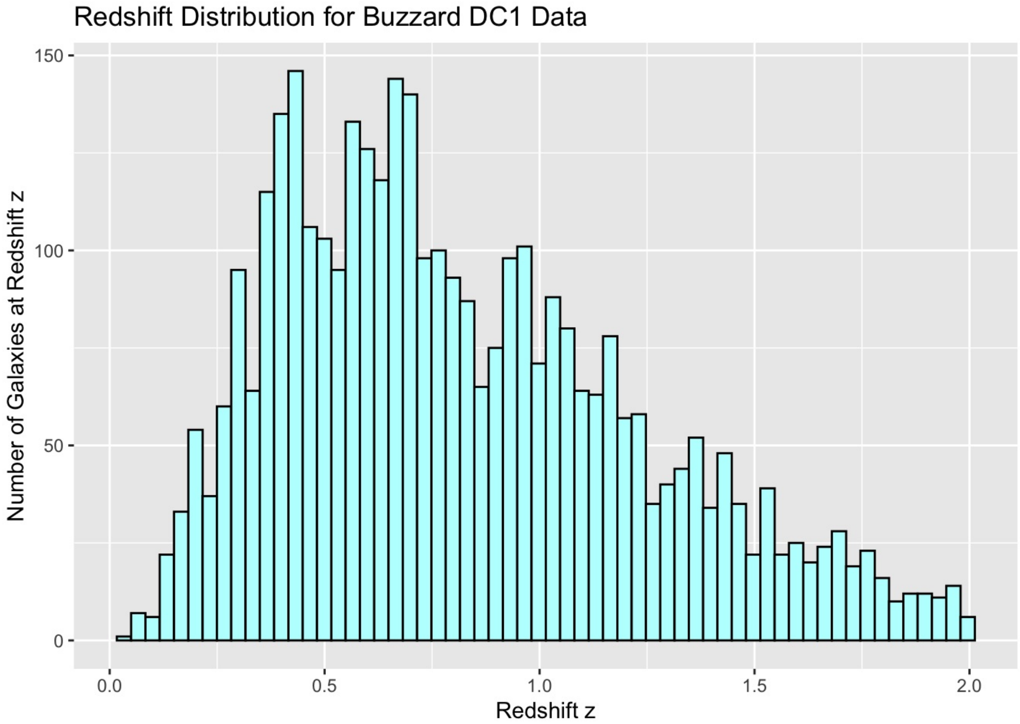



```
modality: unimodal; skewed right; jaggedness due to randomness, noise
```

Question 2

Repeat Q1, but change the x-axis label to “Redshift z” and the y-axis label to “Number of Galaxies at Redshift z”. Also, add a title to the plot: “Redshift Distribution for Buzzard DC1 Data”.

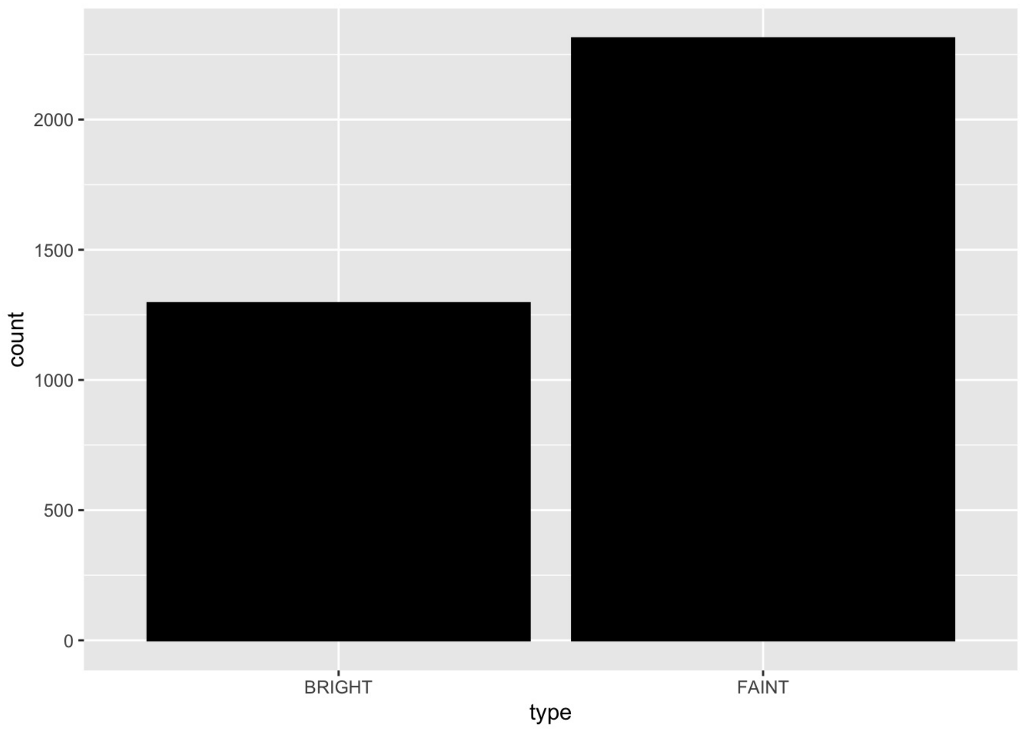
```
ggplot(data=data.frame(response),mapping=aes(x=response)) + geom_histogram(color="black", fill="paleturquoise1",bins=60) + ggtitle("Redshift Distribution for Buzzard DC1 Data") + xlab("Redshift z") + ylab("Number of Galaxies at Redshift z")
```



Question 3

Construct a bar chart that shows the distribution of galaxy `type`, here defined to be `BRIGHT` and `FAINT`. Remember that a bar chart is a representation of a probability mass function, or pmf. As a reminder to yourself, also show the number of counts for each type, coded in some manner that you’ve learned thus far.

```
ggplot(data=data.frame(predictors),mapping=aes(x=type)) + geom_bar(color="black", fill="black")
```



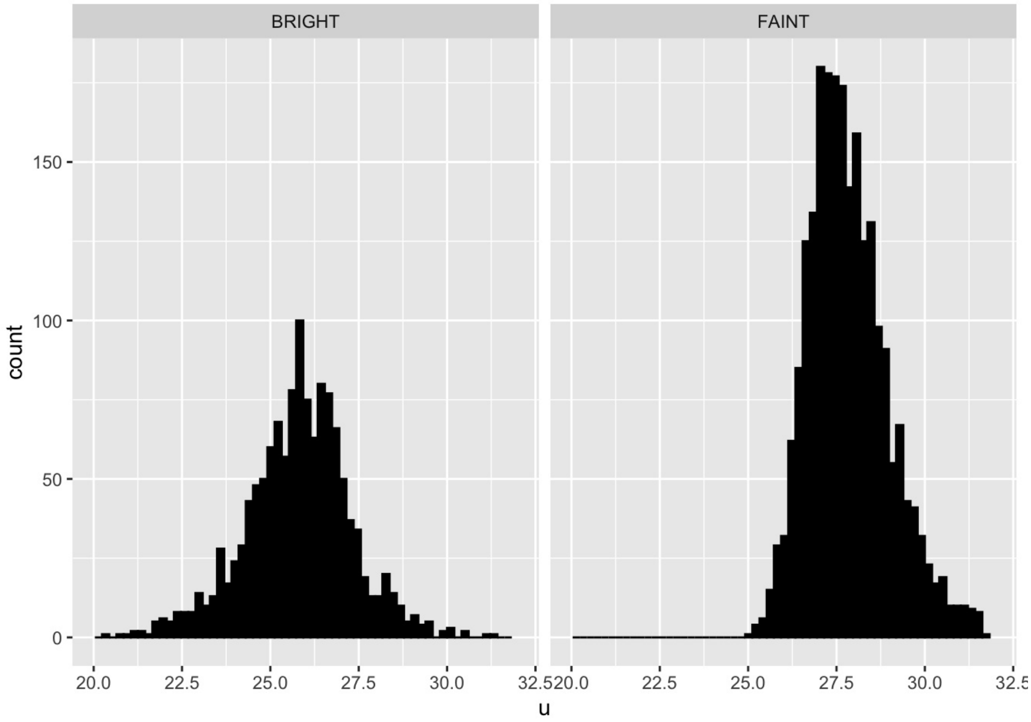
Question 4

Utilize facet wrapping to display two histograms side-by-side: one showing `u` -band magnitudes for `BRIGHT` data, and one showing `u` -band magnitudes for `FAINT` data. Note that to compare distributions, it is helpful *to have the data span the same scale*. So add a function to the string of functions that sets the x-axis limits to be 20 and 32.

```
ggplot(data=data.frame(predictors),mapping=aes(x=u)) + geom_histogram(color="black", fill="black", bins=60) + facet
_wrap(~type,scales='free_x') + xlim(20, 32)
```

```
## Warning: Removed 20 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 4 rows containing missing values (geom_bar).
```

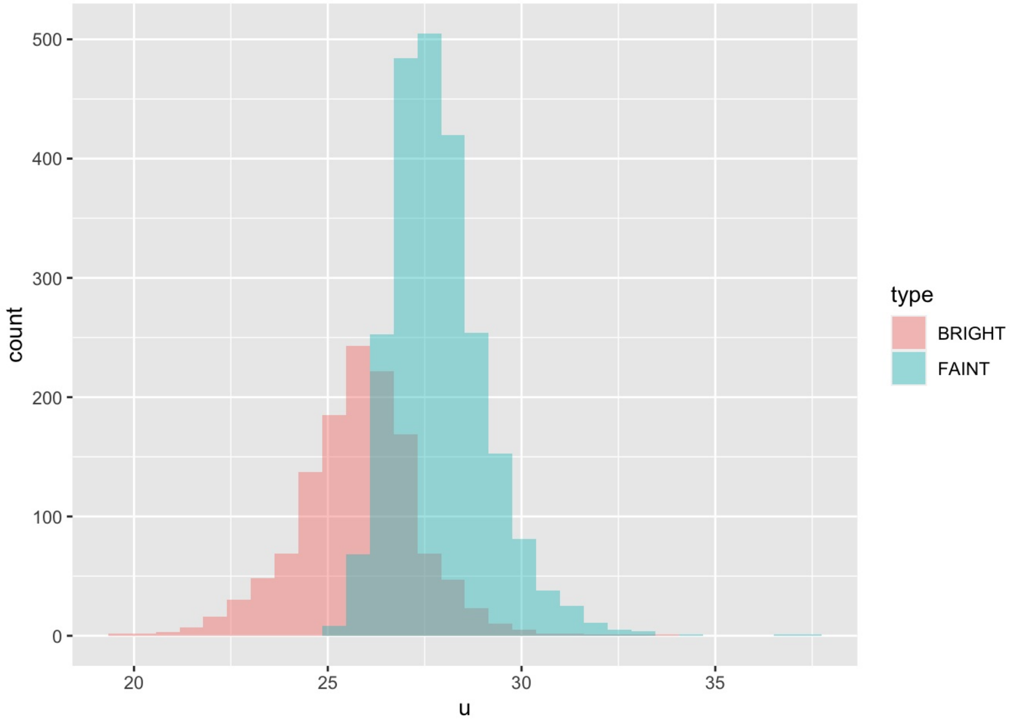


Question 5

Show the same information as in Q4, except use overlapping, partially transparent histograms. Construct such a histogram (again for u). (Hints: in the call to `aes()` in the `ggplot()` call, include `fill=type`, and in the `geom_histogram()` call, include two new arguments: `alpha` and `position="Identity"`. The `alpha` argument controls the transparency: 0 for invisible to 1 for totally opaque.

```
ggplot(data=data.frame(predictors),mapping=aes(x=u, fill=type)) + geom_histogram( alpha=.4, position="Identity")
```

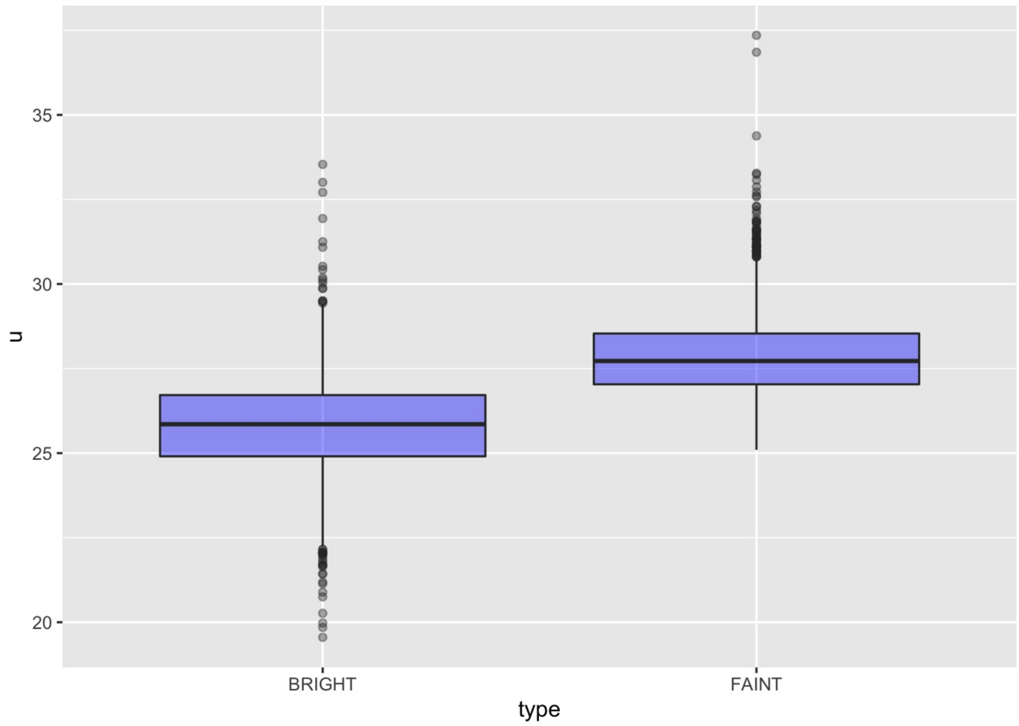
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Question 6

Show the same information as in Q4, except that instead of using side-by-side histograms, use side-by-side boxplots. Unlike in Q4, you do not need to apply `facet_wrap()` . Instead, for the aesthetics, specify that `x` is `type` and `y` is `u` .

```
ggplot(data=data.frame(predictors),mapping=aes(x=type, y=u)) + geom_boxplot(alpha=.4, fill="blue")
```



Question 7

In order to illustrate the `gather()` function, we are going to remove the `type` column from `predictors` :

```
# UNCOMMENT THE LINE BELOW AND RUN CHUNK
pred.notype = predictors[,-1]  # keep all rows, remove first column
```

Apply the `gather()` function to `pred.notype` , and show the first six rows of output by piping to the `head()` function. What are the names of the columns of the data frame output by `gather()` ? (To understand more fully what `gather()` is doing: it is taking a multi-column data frame and reshaping it to have two columns: the first contains the variable name, and the second the variable values.)

```
gather(pred.notype) %>% head()
```

```
##   key   value
## 1   u 23.5907
## 2   u 27.1252
## 3   u 27.4013
## 4   u 27.9354
## 5   u 27.7733
## 6   u 27.1742
```

```
Key and Value
```

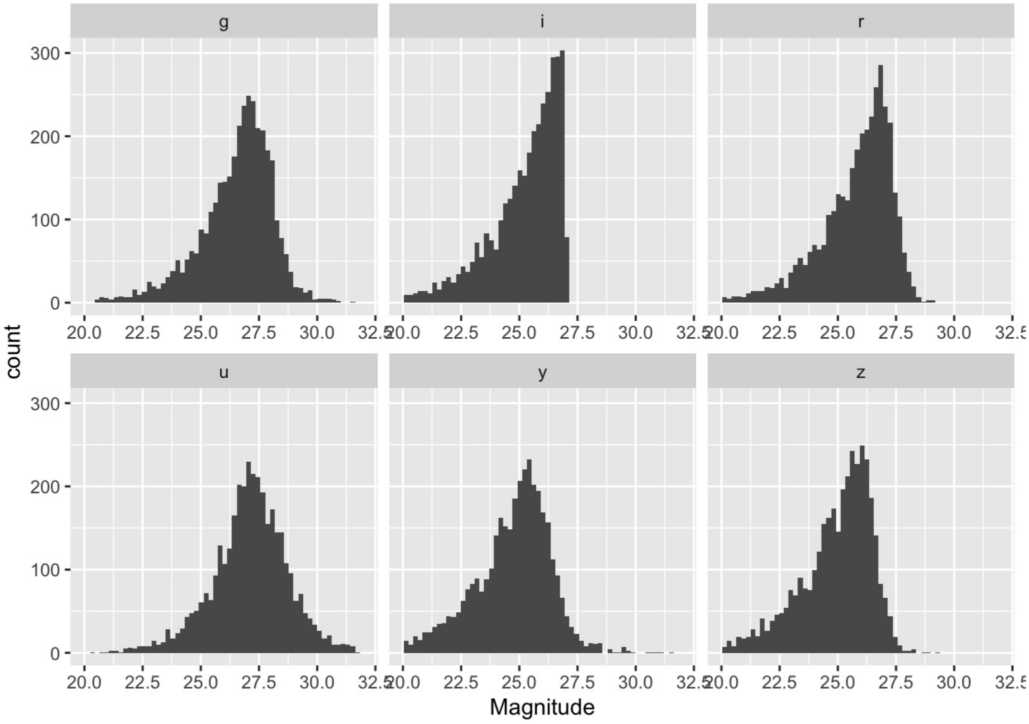
Question 8

Create a faceted histogram that takes `gather(pred.notype)` as the input data frame and `value` as the aesthetic, and wraps on `key` . Set the x-axis limits to be 20 and 32. Replace the x-axis label with “Magnitude”.

```
ggplot(data=gather(pred.notype),mapping=aes(x=value)) + geom_histogram(bins=60) +  facet_wrap(~key,scales='free_x')
+ xlim(20, 32) + xlab("Magnitude")
```

```
## Warning: Removed 230 rows containing non-finite values (stat_bin).
```

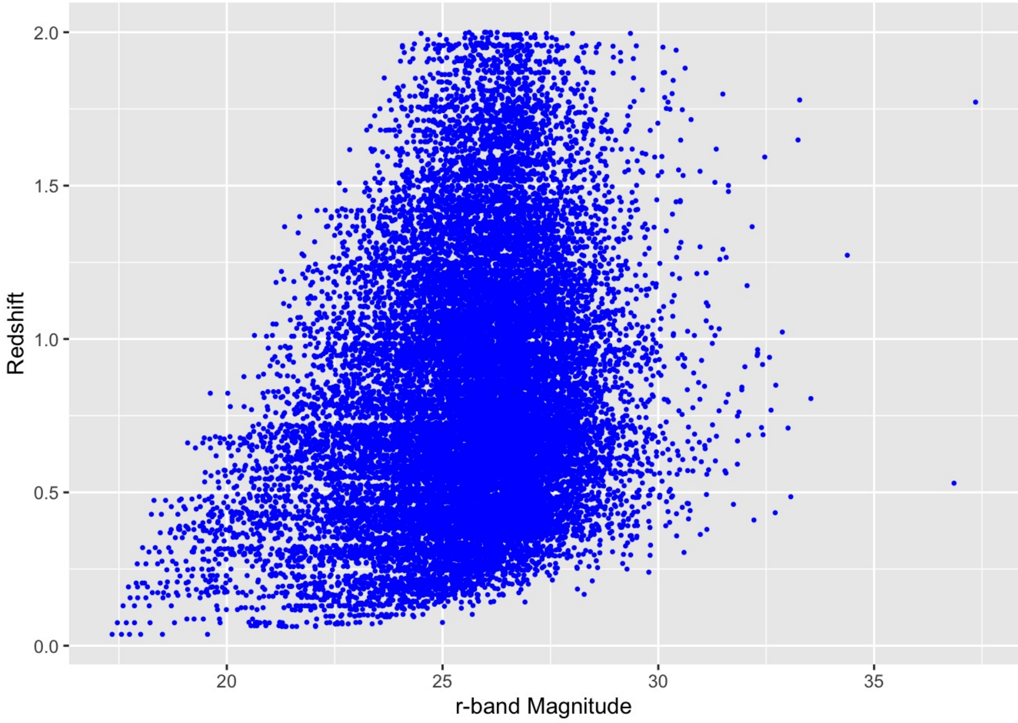
```
## Warning: Removed 12 rows containing missing values (geom_bar).
```



Question 9

Use `geom_point()` to create a scatter plot of redshift vs. r-band magnitude. (Remember, one plots y vs. x, so redshift will go on the y-axis here.) Make the point size 0.5, change the x-axis label to “r-band magnitude”, and change the y-axis label to “Redshift”. Oh, and add some color!

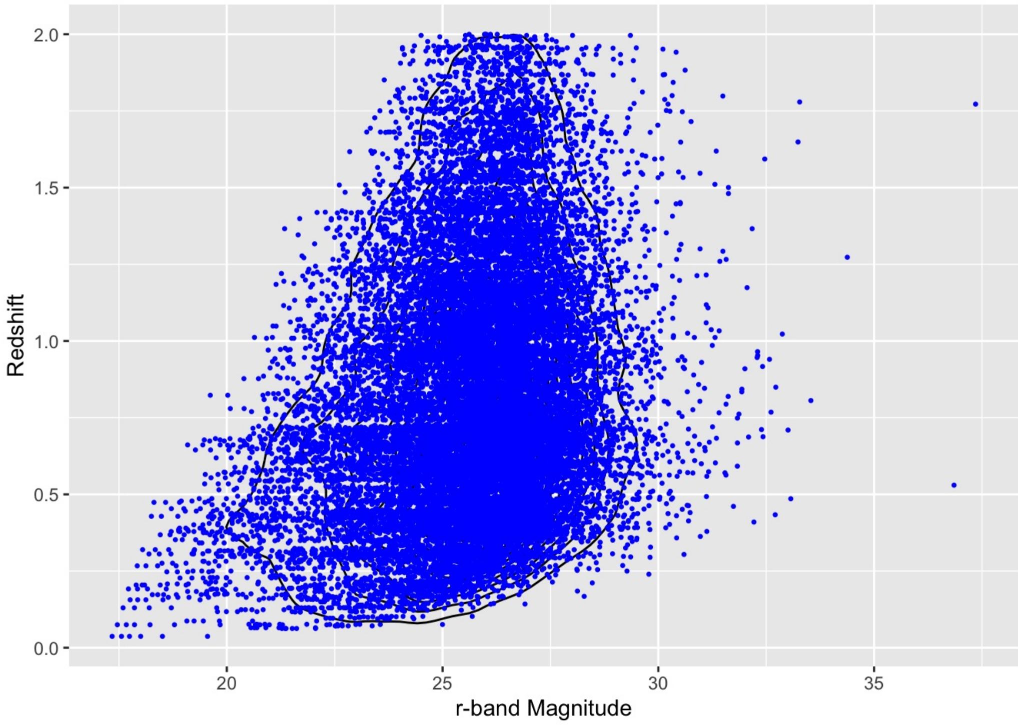
```
df <- cbind(gather(pred.notype), response)


ggplot(data=df,mapping=aes(x=value, y=response)) + geom_point(color="blue", size=.5) + xlab("r-band Magnitude")+ yl
ab("Redshift")
```



Question 10

An issue with scatter plots is that they are not, by themselves, estimates of bivariate distributions; their points are simply samples from such distributions. To try to visualize the underlying distribution, one can do a few things. Here, repeat Q9, and add `geom_density_2d()` so as to overlay a contour plot of the estimated density from which the r-band magnitude and the redshift are sampled.

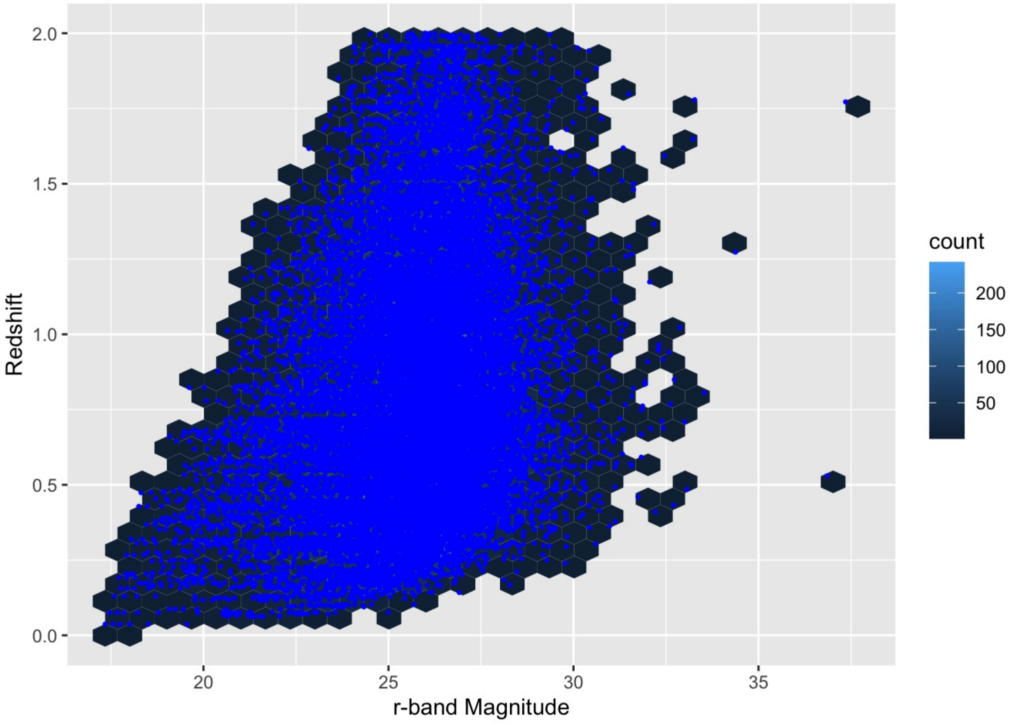
```
ggplot(data=df,mapping=aes(x=value, y=response)) + geom_density_2d(color="black") + geom_point(color="blue", size=.
5) + xlab("r-band Magnitude")+ ylab("Redshift")
```



Question 11

Repeat Q10, but use `geom_hex()` instead. You may have to install the package `hexbin` first!

```
library(hexbin)

ggplot(data=df,mapping=aes(x=value, y=response)) + geom_hex() + geom_point(color="blue", size=.5) + xlab("r-band Ma
gnitude")+ ylab("Redshift")
```



2.0
1.5
1.0
0.5
0.0
Redshift
20
25
30
35
r-band Magnitude
count
200
150
100
50