Math 660

Sep 2023

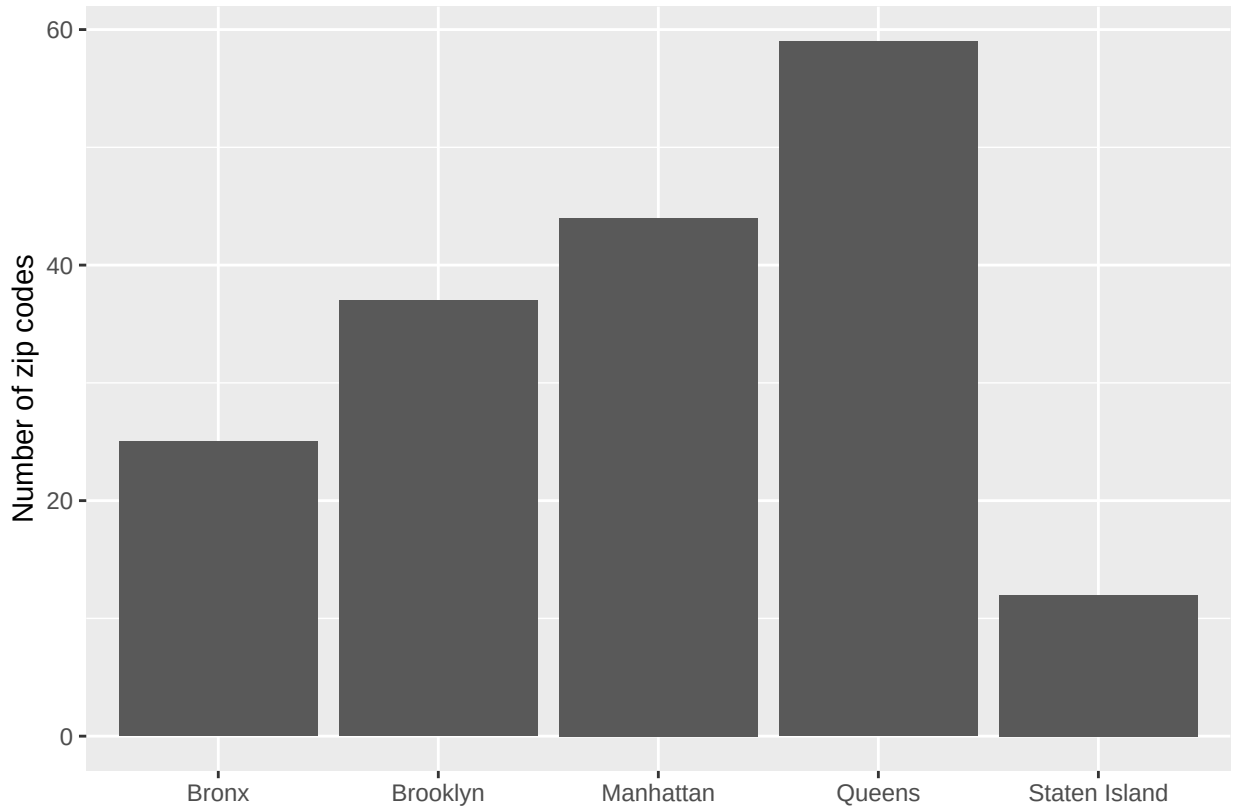
Name: Krisha Shah

1. Use the ggplot2 package to recreate the following plots using the Covid-19 datasets. You are supposed to match the plots shown as closely as possible, including colors, symbols, points/lines, labels etc.

(a) A bar graph showing the number of zipcodes in each NYC borough.

```
qplot(BOROUGH_GROUP, data = zip_code_data,xlab="",ylab="Number of zip codes")
```

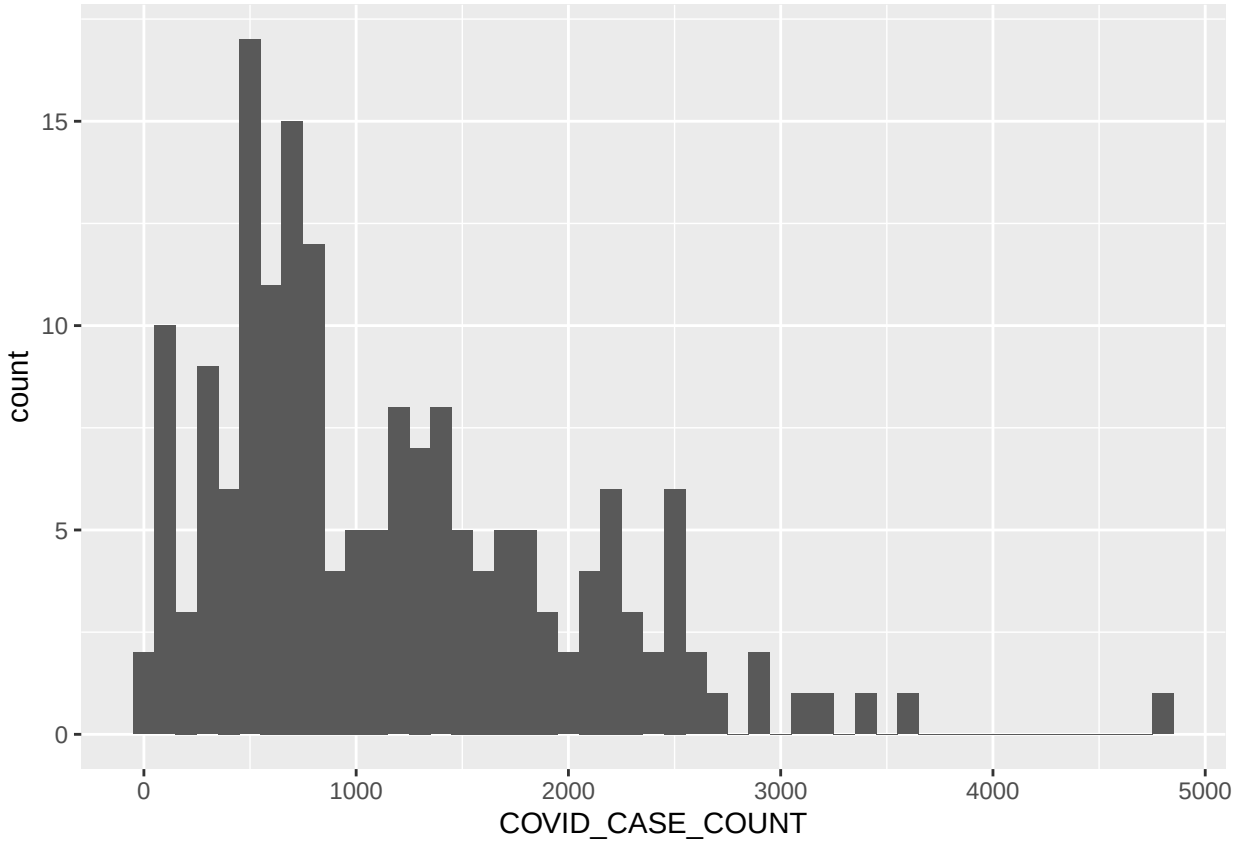
```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Comment: X-axis shows the group of NYC borough and Y-axis shows the count of zip codes. We can conclude that Queens has the maximum number of zip codes.

(b) Histogram of the number of Covid cases by zipcode, using a bin width of 100.

```
qplot(COVID_CASE_COUNT, data = zip_code_data,ylab="count",binwidth = 100)
```

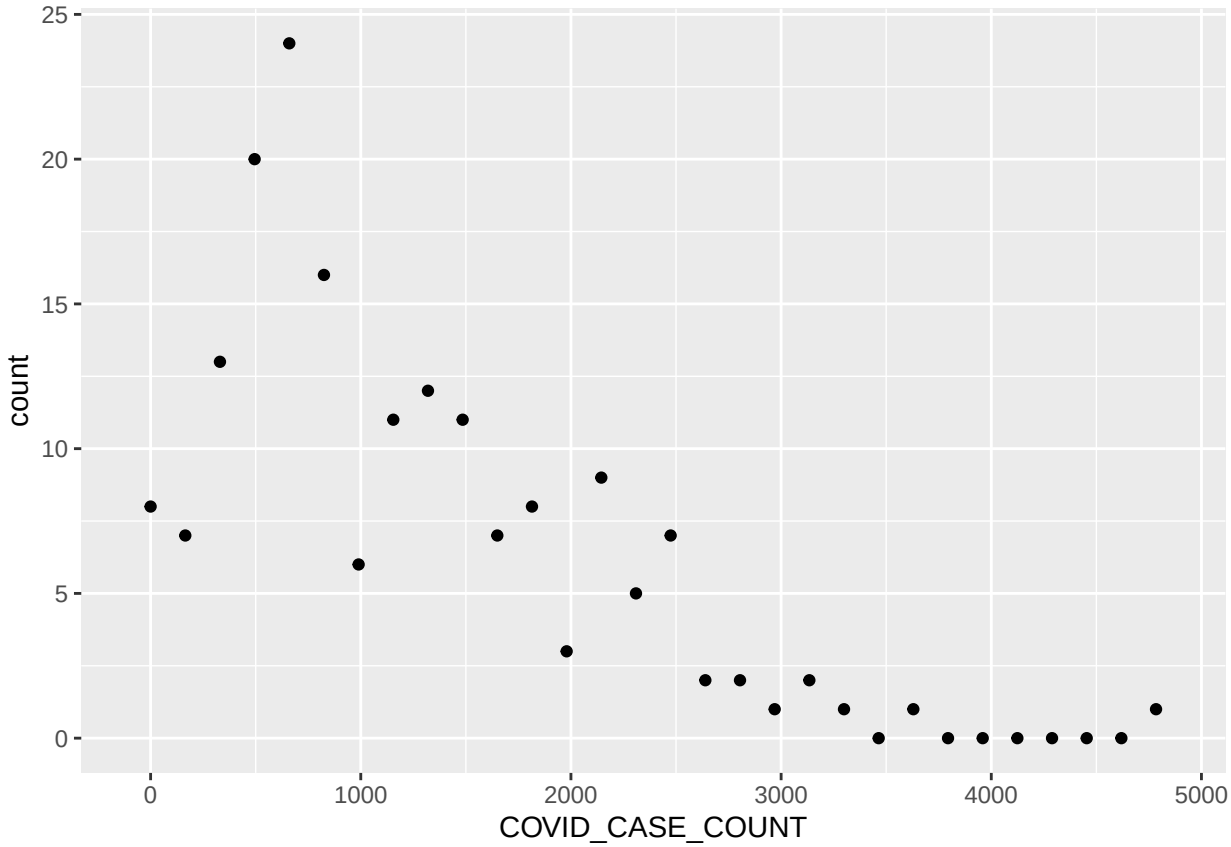


Comment: The histogram displays the distribution of COVID case counts across different zip codes. Each bar in the histogram represents a range of COVID case counts (bin width of 100), and the height of each bar indicates the number of zip codes falling within that particular range.

(c) Similar to a histogram of the number of Covid cases by zipcode, using a bin width of 100, but with the bars represented by points instead.

```
ggplot(zip_code_data, aes(x=COVID_CASE_COUNT)) + stat_bin(geom ="point")
```

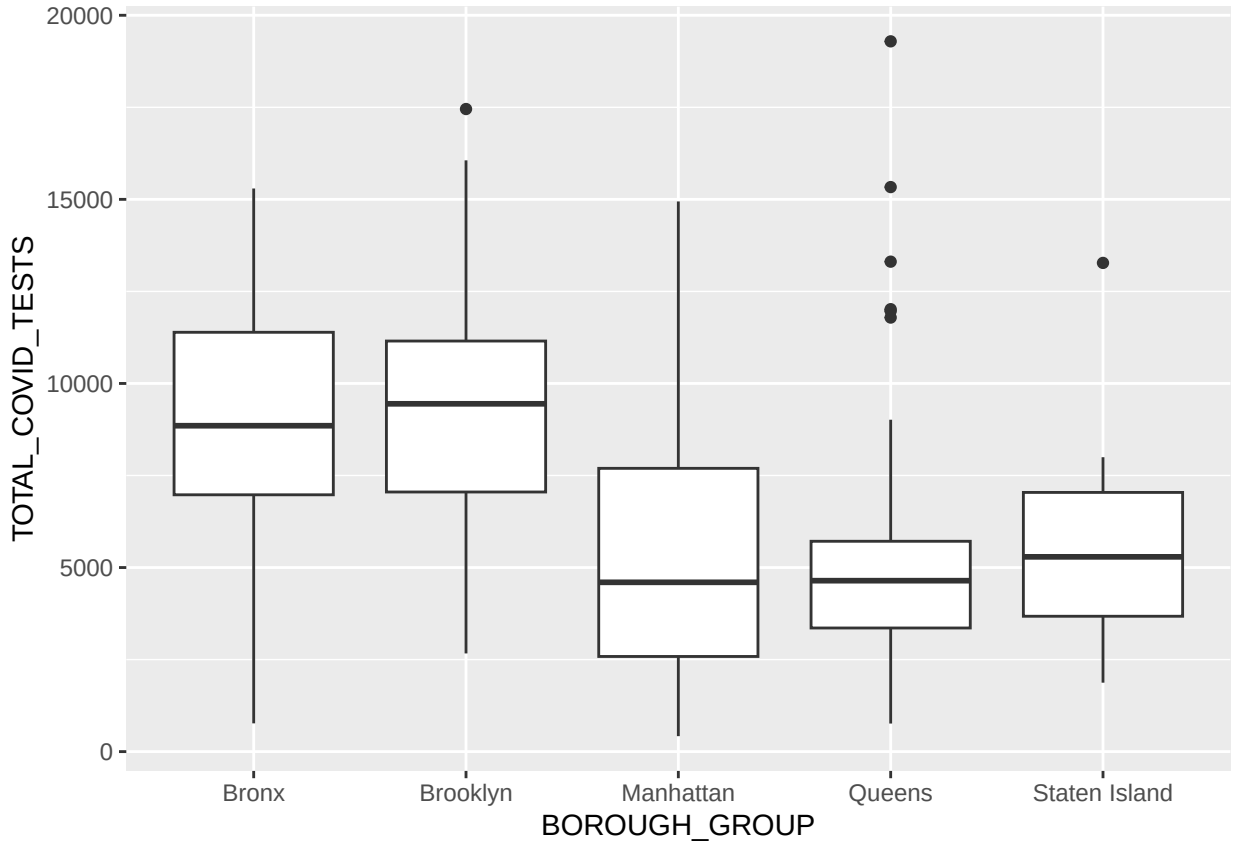
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comment: The plot is similar to previous question. The conclusion is same just that instead of histogram we represent point plot.

(d) Side-by-side boxplots of the total number of tests conducted in each zipcode

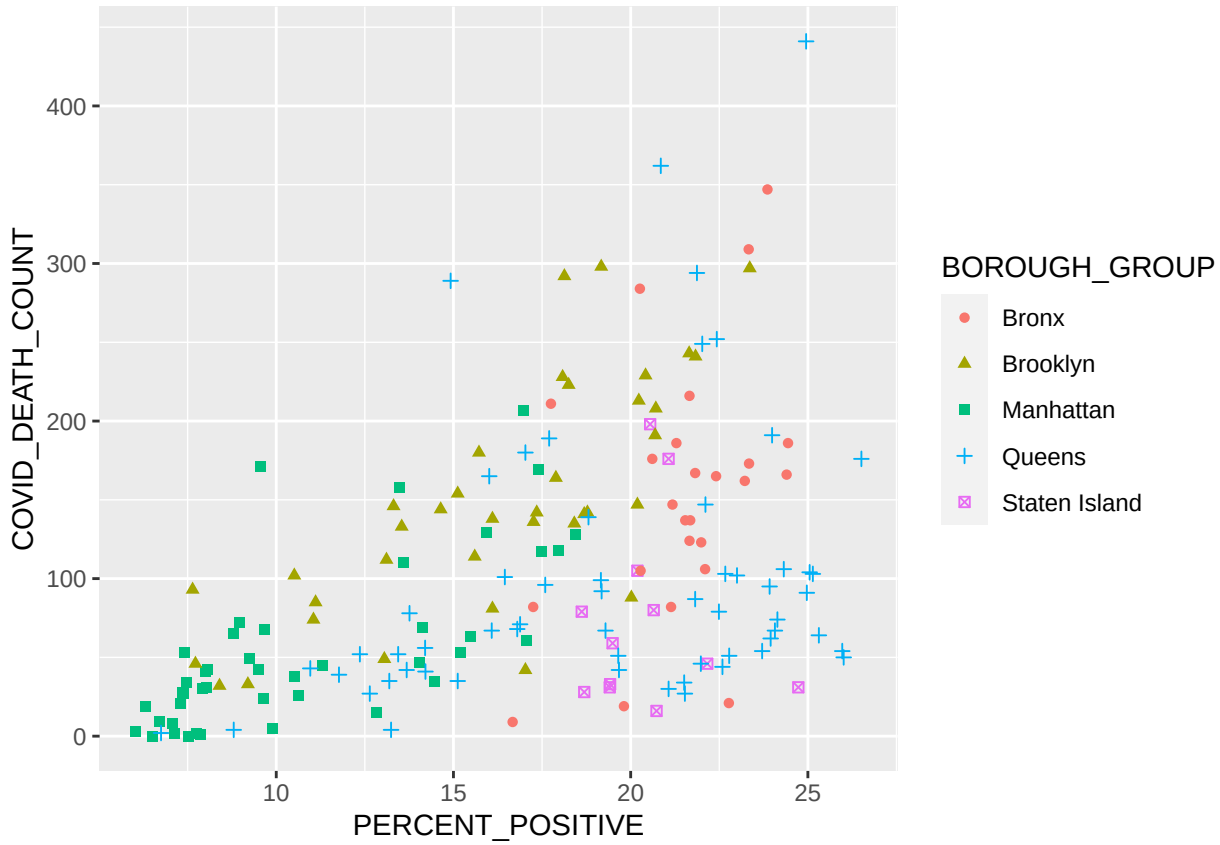
```
qplot(factor(BOROUGH_GROUP),TOTAL_COVID_TESTS, data = zip_code_data, geom ="boxplot", xlab="BOROUGH_GROU
```



Comment: The side-by-side boxplots provide a visual representation of the distribution of total COVID tests conducted in different zip codes, categorized by borough group. We can easily compare the distribution of total COVID tests conducted across borough groups which allows to identify variations in testing patterns and potential outliers in specific boroughs.

(e) Scatter plot of Covid deaths by zipcode vs Percent positive by zipcode

```
qplot(PERCENT_POSITIVE, COVID_DEATH_COUNT, data= zip_code_data, colour = BOROUGH_GROUP, shape = BOROUGH_
```

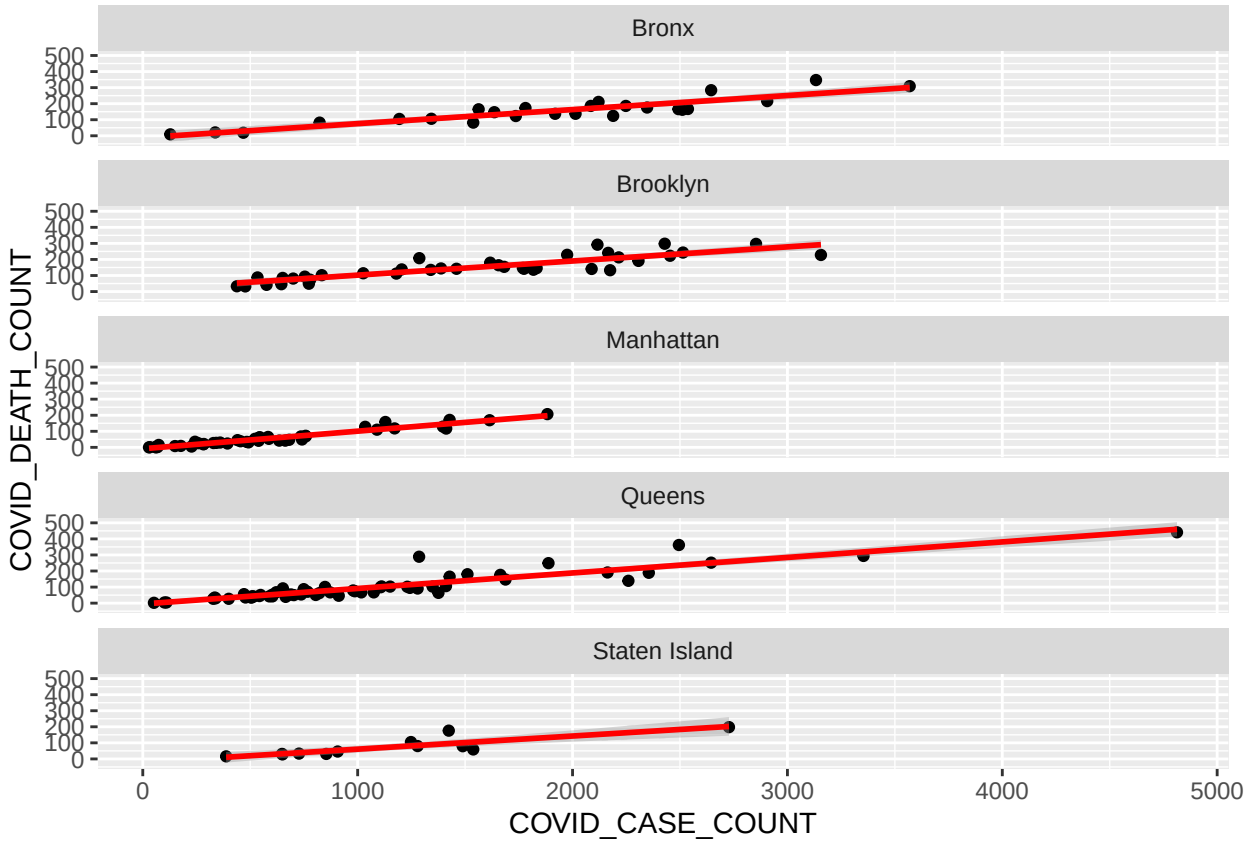


Comment: The scatter plot shows percent of positive cases on the X-axis and covid death count on the Y-axis. Here we try to show different borough groups using various shapes and symbols.

(f) Scatter plot of zipcode deaths vs case counts, separated by borough and with a fitted line

```
qplot(COVID_CASE_COUNT, COVID_DEATH_COUNT, data= zip_code_data, facets = ~BOROUGH_GROUP) + geom_smooth(
```

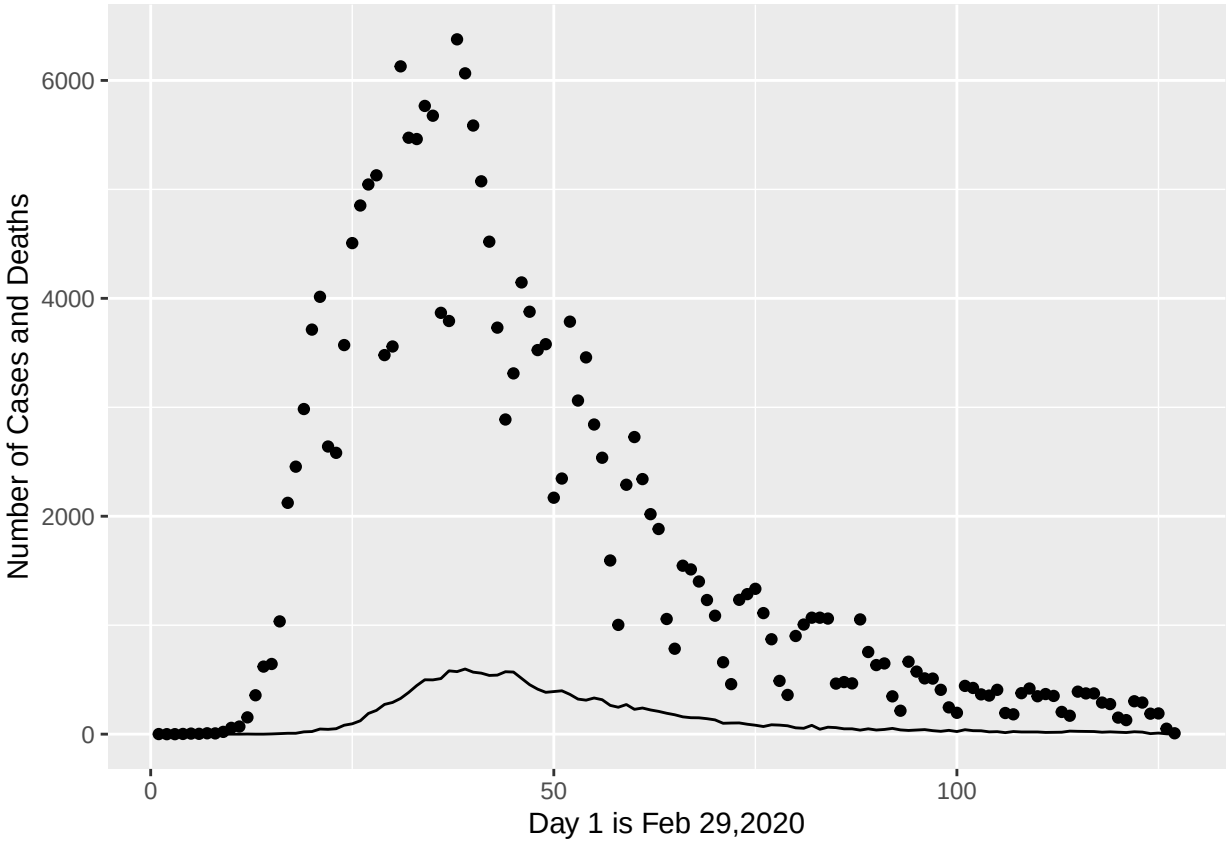
```
## `geom_smooth()` using formula = 'y ~ x'
```



Comment: We have created a scatter plot here for covid case counts vs covid dead counts. The linear fitted line is created using geom_smooth and to showcase all borough graphs in a single plot we use facets.

(g) A timeplot of the daily number of cases and number of deaths

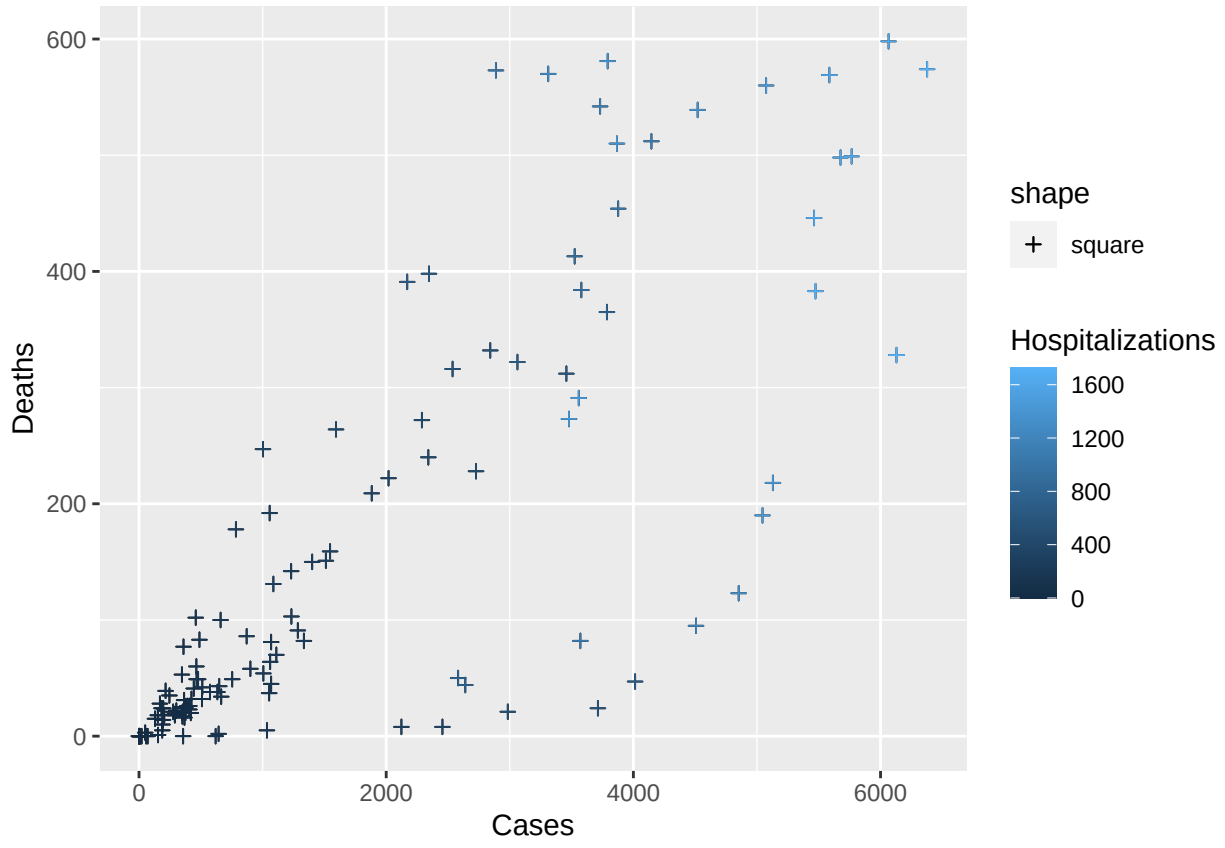
```
qplot(y=Cases, data= daily_data, xlab="Day 1 is Feb 29,2020", ylab="Number of Cases and Deaths") + geom_
```



Comment: The following plot shows scatter dots to represent number of cases and the solid line represents number of deaths using geom_line. And the data is starting from Feb 29, 2020.

(h) Scatter plot of the daily number of deaths vs daily number of cases

```
qplot(x= Cases, y=Deaths, data= daily_data, shape = "square") + geom_point(aes(colour = Hospitalizations
```



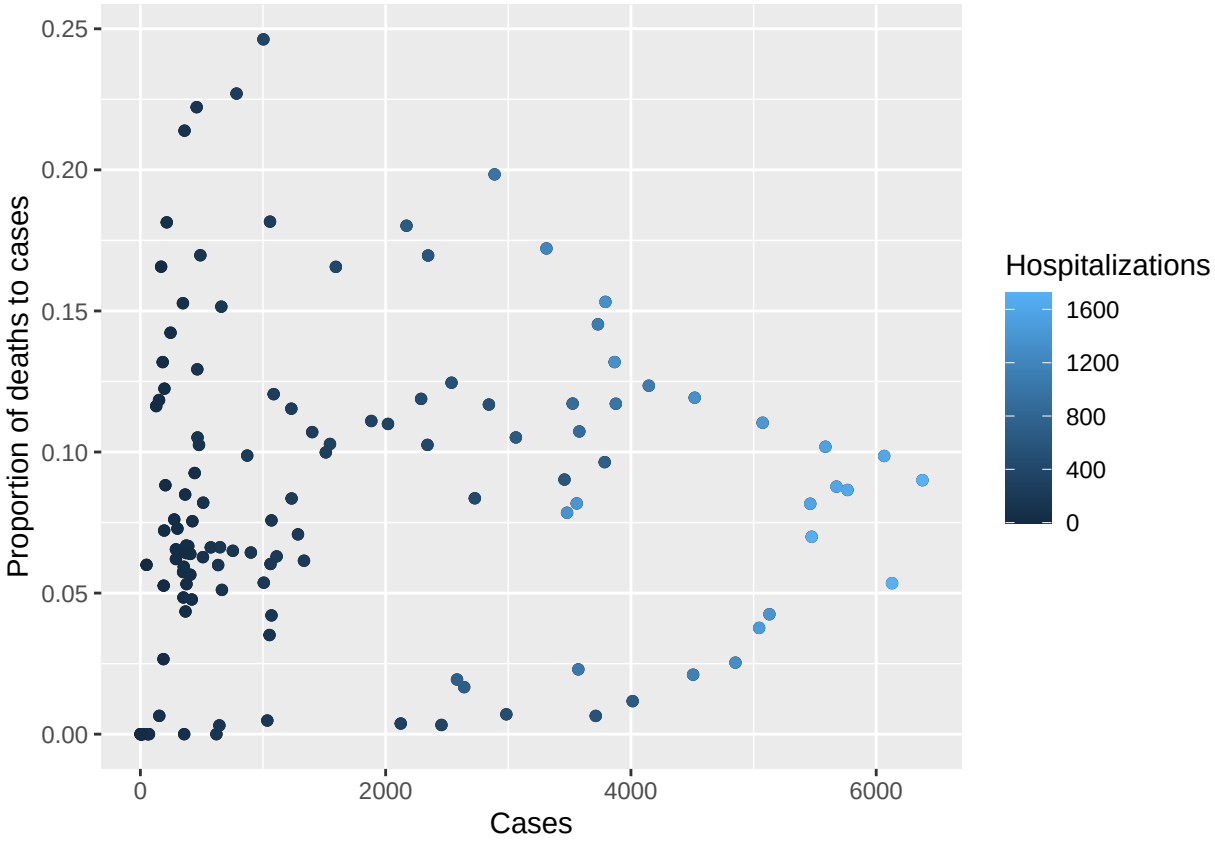

```
# Reference: https://ggplot2.tidyverse.org/
```

Comment: This is a normal scatter plot like any other previous questions. However we have added a geom_point color which automatically sets shades of blue and a plus symbol using scale_shape_manual with a value 3 that is for the plus sign.

(i) Scatter plot of the daily proportion of deaths to cases, vs number of cases

```
qplot(y= Deaths/Cases, x=Cases, data= daily_data, ylab="Proportion of deaths to cases") + geom_point(ae
```

```
## Warning: Removed 2 rows containing missing values (`geom_point()`).
## Removed 2 rows containing missing values (`geom_point()`).
```



Comment: In this scatter plot, we show proportion of deaths to cases in the Y-axis and number of cases on the X-axis. The color coding is again set using geom_point(aes(color)).

2. Write R code to perform the following with the Covid data

(a) Find the total number of cases in New York City. Find also the total population.

```
total_cases = sum(zip_code_data$COVID_CASE_COUNT)
cat("Total number of cases in New York City =", total_cases)
```

```
## Total number of cases in New York City = 207461
```

```
total_population = sum(floor(zip_code_data$POP_DENOMINATOR))
cat("\nTotal population =", total_population)
```

```
##
## Total population = 8394268
```

(b) Obtain summary statistics of all the numerical columns in Covidbyzip. Use the lapply and sapply functions

```r
df <- zip_code_data[, c("COVID_CASE_COUNT", "COVID_CASE_RATE", "POP_DENOMINATOR", "COVID_DEATH_COUNT",

lapply(df, summary)
```

```
## $COVID_CASE_COUNT
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      29     543     912    1172    1664    4813
##
## $COVID_CASE_RATE
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   634.2  1699.5  2479.9  2390.6  3015.9  4563.6
##
## $POP_DENOMINATOR
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3458   26614   43030   47426   67089  111594
##
## $COVID_DEATH_COUNT
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    42.0    82.0   104.7   147.0   441.0
##
## $COVID_DEATH_RATE
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   134.7   197.6   207.2   262.6   708.9
##
## $PERCENT_POSITIVE
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.04   13.11   17.89   16.94   21.65   26.51
##
## $TOTAL_COVID_TESTS
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     420    3697    5889    6614    8918   19292
```

```r
sapply(df, summary)
```

```
##         COVID_CASE_COUNT COVID_CASE_RATE POP_DENOMINATOR COVID_DEATH_COUNT
## Min.              29.000         634.180         3457.77            0.0000
## 1st Qu.          543.000        1699.470        26614.42           42.0000
## Median           912.000        2479.950        43030.43           82.0000
## Mean            1172.096        2390.598        47425.73          104.7345
## 3rd Qu.         1664.000        3015.870        67089.29          147.0000
## Max.            4813.000        4563.630       111594.10          441.0000
##         COVID_DEATH_RATE PERCENT_POSITIVE TOTAL_COVID_TESTS
## Min.             0.0000          6.04000           420.000
## 1st Qu.        134.7200         13.11000          3697.000
## Median         197.6000         17.89000          5889.000
## Mean           207.1556         16.93921          6613.644
## 3rd Qu.        262.5500         21.65000          8918.000
## Max.           708.9100         26.51000         19292.000
```

(c) Obtain summary statistics of Covid case counts for Manhattan zipcodes only.

```
covid_case_count1 <- zip_code_data[zip_code_data$BOROUGH_GROUP == "Manhattan","COVID_CASE_COUNT"]
covid_case_count_summary1 <- summary(covid_case_count1)
print("For Manhattan Zipcode:")
```

```
## [1] "For Manhattan Zipcode:"
```

```
print(covid_case_count_summary1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    29.0   254.2   507.0   585.4   745.2  1883.0
```

(d) Do the same thing for the Queens zipcodes only and Staten Island zipcodes only.

```
covid_case_count2 <- zip_code_data[zip_code_data$BOROUGH_GROUP == "Queens","COVID_CASE_COUNT"]
covid_case_count_summary2 <- summary(covid_case_count2)
print("For Queens Zipcode:")
```

```
## [1] "For Queens Zipcode:"
```

```
print(covid_case_count_summary2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    52.0   595.5   848.0  1083.3  1317.5  4813.0
```

```
covid_case_count3 <- zip_code_data[zip_code_data$BOROUGH_GROUP == "Staten Island","COVID_CASE_COUNT"]
covid_case_count_summary3 <- summary(covid_case_count3)
print("For Staten Island Zipcode:")
```

```
## [1] "For Staten Island Zipcode:"
```

```
print(covid_case_count_summary3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   388.0   707.8  1077.5  1156.7  1440.2  2728.0
```

(e) Identify the NYC zipcode and neighborhood name with the highest case rate.

```
highest_zipcode <- zip_code_data[which.max(zip_code_data$COVID_CASE_RATE),c(1,2)]
print(highest_zipcode$NEIGHBORHOOD_NAME)
```

```
## [1] "Airport/East Elmhurst"
```

(f) On which day were there the most number of deaths relative to hospitalizations in NY State

```r
Most_Deaths <- daily_data$Deaths/daily_data$Hospitalizations

max(subset(Most_Deaths, Most_Deaths>=0 & Most_Deaths<Inf))
```

```
## [1] 1.428571
```

3. Write R code to do the following:

(a) Use seq to create a vector with elements 2, 4, 6, ... to 24. Extract the 5th value of this vector

```r
vector1 <- seq(2, 24, by=2)
cat("The 5th value of this vector is: ", vector1[5])
```

```
## The 5th value of this vector is:  10
```

(b) Reverse the order of your vector above.

```r
print(rev(vector1))
```

```
##  [1] 24 22 20 18 16 14 12 10  8  6  4  2
```

(c) Create a vector consisting of, fifty 1's, twenty-six 2's and seventeen 3's, in that order.

```r
vector2 <- rep(c(1,2,3), times=c(50,26,17))
print(vector2)
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [77] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

(d) Create a diagonal matrix with the numbers 1, 2, 3, ... 10 in the diagonal. Find its inverse

```r
diag_matrix <- diag(seq(1:10))
print(diag_matrix)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    2    0    0    0    0    0    0    0     0
##  [3,]    0    0    3    0    0    0    0    0    0     0
##  [4,]    0    0    0    4    0    0    0    0    0     0
##  [5,]    0    0    0    0    5    0    0    0    0     0
##  [6,]    0    0    0    0    0    6    0    0    0     0
```

```
##  [7,]    0    0    0    0    0    0    7    0    0     0
##  [8,]    0    0    0    0    0    0    0    8    0     0
##  [9,]    0    0    0    0    0    0    0    0    9     0
## [10,]    0    0    0    0    0    0    0    0    0    10
```

```
inverse_matrix <- solve(diag_matrix)
print(inverse_matrix)
```

```
##       [,1] [,2]      [,3] [,4] [,5]      [,6]      [,7]  [,8]      [,9] [,10]
##  [1,]    1  0.0 0.0000000 0.00  0.0 0.0000000 0.0000000 0.000 0.0000000   0.0
##  [2,]    0  0.5 0.0000000 0.00  0.0 0.0000000 0.0000000 0.000 0.0000000   0.0
##  [3,]    0  0.0 0.3333333 0.00  0.0 0.0000000 0.0000000 0.000 0.0000000   0.0
##  [4,]    0  0.0 0.0000000 0.25  0.0 0.0000000 0.0000000 0.000 0.0000000   0.0
##  [5,]    0  0.0 0.0000000 0.00  0.2 0.0000000 0.0000000 0.000 0.0000000   0.0
##  [6,]    0  0.0 0.0000000 0.00  0.0 0.1666667 0.0000000 0.000 0.0000000   0.0
##  [7,]    0  0.0 0.0000000 0.00  0.0 0.0000000 0.1428571 0.000 0.0000000   0.0
##  [8,]    0  0.0 0.0000000 0.00  0.0 0.0000000 0.0000000 0.125 0.0000000   0.0
##  [9,]    0  0.0 0.0000000 0.00  0.0 0.0000000 0.0000000 0.000 0.1111111   0.0
## [10,]    0  0.0 0.0000000 0.00  0.0 0.0000000 0.0000000 0.000 0.0000000   0.1
```

(e) Extract the 5th column of the matrix you created in 3(d). Make this vector (i.e the 5th column) into a 2×5 matrix, fillng out the matrix by row.

```
new_matrix <- inverse_matrix[,5]
print(matrix(new_matrix, nrow = 2, byrow = TRUE))
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0  0.2
## [2,]    0    0    0    0  0.0
```