Lab 2: Mapping with ggplot

Jeremy R. Porter

Spring 2020

####In today's lab we are going to use the internal maping facilities from the `maps` package to map Quality of Life at the county level in the state of New York.

####First, let's set our working directory. . . .

```
knitr::opts_chunk$set(
    echo = TRUE,
    message = TRUE,
    warning = TRUE
)
setwd("C:/Prop_val2/stuff")
```

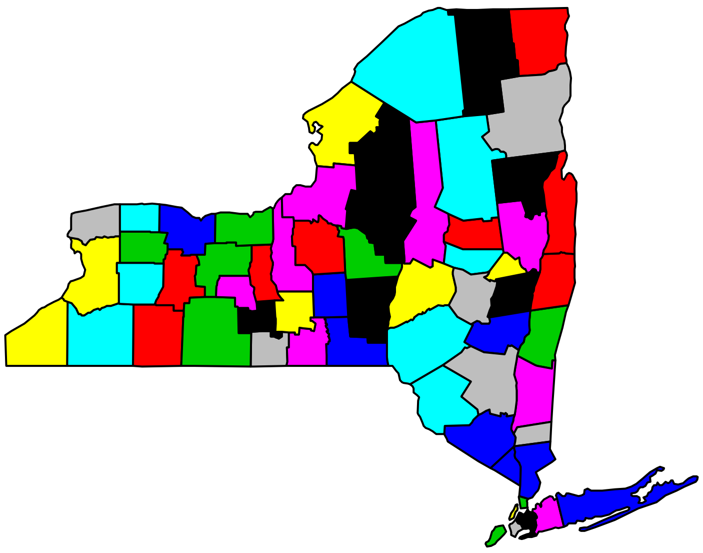
####Next, let's require the `maps` pacakage (Note: for any package not already installed in your local environment, you should first `install.packages("[Package Name]")`)

```
require(maps)
```

```
## Loading required package: maps
```

####Let's pull the county level polygon map from the `maps` package. The option `fill=TRUE` gives us a polygon as opposed to a line "map". Next, I am going to check the new map using the `head` and the `length` function to explore the existence of duplicated county names for the joins that I will implement later.

```
ny_cty <- map('county', 'new york', fill=TRUE, col = palette())
```



```
length(!duplicated(ny_cty$names))
```

```
## [1] 62
```

####To create IDs for the map object, I first need to create a list of county names and I am going to separate the county name from the state name by using the `strsplit` function to split the `ny_cty$names` variable at the comma `","`. After slpitting the variable I am going to assign the `map.IDs` to the lowercase version of the 2nd column of the `list.names.ny` object (which is the actual county name). Creating a lower case version of the file simply standardizes the names so that I can easily join data to the object. Another good idea in the standardization of the data is to remove spaces and symbols (periods in names like St. Lawrence for instance), which I do with the `gsub` command.

```
list.names.ny <- strsplit(ny_cty$names,",")
head(list.names.ny)
```

```
## [[1]]
## [1] "new york" "albany"
##
## [[2]]
## [1] "new york" "allegany"
##
## [[3]]
## [1] "new york" "bronx"
##
## [[4]]
## [1] "new york" "broome"
##
## [[5]]
## [1] "new york"    "cattaraugus"
##
## [[6]]
## [1] "new york" "cayuga"
```

```
View(list.names.ny)

map.IDs <- as.character(tolower(sapply(list.names.ny, function(x) x[2])))
head(map.IDs, n=62)
```

```
##  [1] "albany"      "allegany"    "bronx"       "broome"      "cattaraugus"
##  [6] "cayuga"      "chautauqua"  "chemung"     "chenango"    "clinton"
## [11] "columbia"    "cortland"    "delaware"    "dutchess"    "erie"
## [16] "essex"       "franklin"    "fulton"      "genesee"     "greene"
## [21] "hamilton"    "herkimer"    "jefferson"   "kings"       "lewis"
## [26] "livingston"  "madison"     "monroe"      "montgomery"  "nassau"
## [31] "new york"    "niagara"     "oneida"      "onondaga"    "ontario"
## [36] "orange"      "orleans"     "oswego"      "otsego"      "putnam"
## [41] "queens"      "rensselaer"  "richmond"    "rockland"    "st lawrence"
## [46] "saratoga"    "schenectady" "schoharie"   "schuyler"    "seneca"
## [51] "steuben"     "suffolk"     "sullivan"    "tioga"       "tompkins"
## [56] "ulster"      "warren"      "washington"  "wayne"       "westchester"
## [61] "wyoming"     "yates"
```

```
map.IDs <- gsub("st lawrence", "stlawrence", map.IDs)
head(map.IDs, n=62)
```

```
##  [1] "albany"      "allegany"    "bronx"       "broome"      "cattaraugus"
##  [6] "cayuga"      "chautauqua"  "chemung"     "chenango"    "clinton"
## [11] "columbia"    "cortland"    "delaware"    "dutchess"    "erie"
## [16] "essex"       "franklin"    "fulton"      "genesee"     "greene"
## [21] "hamilton"    "herkimer"    "jefferson"   "kings"       "lewis"
## [26] "livingston"  "madison"     "monroe"      "montgomery"  "nassau"
## [31] "new york"    "niagara"     "oneida"      "onondaga"    "ontario"
## [36] "orange"      "orleans"     "oswego"      "otsego"      "putnam"
## [41] "queens"      "rensselaer"  "richmond"    "rockland"    "stlawrence"
## [46] "saratoga"    "schenectady" "schoharie"   "schuyler"    "seneca"
## [51] "steuben"     "suffolk"     "sullivan"    "tioga"       "tompkins"
## [56] "ulster"      "warren"      "washington"  "wayne"       "westchester"
## [61] "wyoming"     "yates"
```

####Now that I have the IDs created, I can use the `maptools` package to transform the map object to a Spatial Polygons object using the `map2SpatialPolygons` function and specifying a projection for central NY (`proj4string = CRS("+init=epsg:2261")`).

```
require(maptools)
```

```
## Loading required package: maptools
```

```
## Loading required package: sp
```

```
## Checking rgeos availability: FALSE
##      Note: when rgeos is not available, polygon geometry     computations in maptools depend on gpcli
##      which has a restricted licence. It is disabled by default;
##      to enable gpclib, type gpclibPermit()
```

```
ny_cty_sp <- map2SpatialPolygons(ny_cty, IDs = map.IDs, proj4string = CRS("+init=epsg:2261"))
head(map.IDs, n=62)
```

```
##  [1] "albany"       "allegany"     "bronx"        "broome"       "cattaraugus"
##  [6] "cayuga"       "chautauqua"   "chemung"      "chenango"     "clinton"
## [11] "columbia"     "cortland"     "delaware"     "dutchess"     "erie"
## [16] "essex"        "franklin"     "fulton"       "genesee"      "greene"
## [21] "hamilton"     "herkimer"     "jefferson"    "kings"        "lewis"
## [26] "livingston"   "madison"      "monroe"       "montgomery"   "nassau"
## [31] "new york"     "niagara"      "oneida"       "onondaga"     "ontario"
## [36] "orange"       "orleans"      "oswego"       "otsego"       "putnam"
## [41] "queens"       "rensselaer"   "richmond"     "rockland"     "stlawrence"
## [46] "saratoga"     "schenectady"  "schoharie"    "schuyler"     "seneca"
## [51] "steuben"      "suffolk"      "sullivan"     "tioga"        "tompkins"
## [56] "ulster"       "warren"       "washington"   "wayne"        "westchester"
## [61] "wyoming"      "yates"
```

####To map the Quality of Life indicator, I have given you a file (`rwj_rank.csv`), which contains a variable `QL.Rank` for the ranking of QUantity of Life within the state (`1=best, 62=worst for NY [62 counties]`). We can `subset` the data to include only NY counties and again, change the county names to lower case (`tolower`) and remove all spaces and symbols using the `gsub` function. Finally, we can set the `row.names` to the lower case version of the county names for the next step.

```
require(data.table)
```

```
## Loading required package: data.table
```

```
rwj <- fread("rwj_rank.csv", stringsAsFactors = F, data.table = F, colClasses=list(character=c("FIPS"))
head(rwj)
```

```
##    FIPS   State  County LL.Rank LL.Quartile QL.Rank QL.Quartile HB.Rank
## 1 01001 Alabama Autauga      18           2       5           1      12
## 2 01003 Alabama Baldwin       4           1       4           1       3
## 3 01005 Alabama Barbour      14           1      49           3      57
## 4 01007 Alabama    Bibb      53           4      25           2      39
## 5 01009 Alabama  Blount      17           1      12           1      11
## 6 01011 Alabama Bullock      58           4      61           4      66
##   HB.Quartile CC.Rank CC.Quartile SE.Rank SE.Quartile PE.Rank PE.Quartile
## 1           1      15           1       3           1      51           4
## 2           1       7           1       8           1      14           1
## 3           4      20           2      58           4      16           1
## 4           3      42           3      42           3      29           2
## 5           1      41           3      13           1      54           4
## 6           4      54           4      62           4       3           1
```

```
ny_rwj <- subset(rwj, State == "New York")
head(ny_rwj)
```

```
##       FIPS    State      County LL.Rank LL.Quartile QL.Rank QL.Quartile
## 1829 36001 New York      Albany      21           2      35           3
## 1830 36003 New York    Allegany      25           2      39           3
## 1831 36005 New York       Bronx      44           3      62           4
## 1832 36007 New York      Broome      52           4      53           4
## 1833 36009 New York Cattaraugus      58           4      60           4
## 1834 36011 New York      Cayuga      33           3      11           1
##      HB.Rank HB.Quartile CC.Rank CC.Quartile SE.Rank SE.Quartile PE.Rank
## 1829      13           1       7           1       9           1      36
## 1830      45           3      48           4      48           4      29
## 1831      40           3      62           4      62           4      59
## 1832      35           3      16           1      47           4       8
## 1833      59           4      54            4      45            3      50
## 1834      55           4      43            3      29            2      47
##       PE.Quartile
## 1829            3
## 1830            2
## 1831            4
## 1832            1
## 1833            4
## 1834            4
```

```
ny_rwj$County <- gsub("St. Lawrence", "stlawrence", ny_rwj$County)
head(ny_rwj$County, n=62)
```

```
##  [1] "Albany"       "Allegany"     "Bronx"        "Broome"       "Cattaraugus"
##  [6] "Cayuga"       "Chautauqua"   "Chemung"      "Chenango"     "Clinton"
## [11] "Columbia"     "Cortland"     "Delaware"     "Dutchess"     "Erie"
## [16] "Essex"        "Franklin"     "Fulton"       "Genesee"      "Greene"
## [21] "Hamilton"     "Herkimer"     "Jefferson"    "Kings"        "Lewis"
## [26] "Livingston"   "Madison"      "Monroe"       "Montgomery"   "Nassau"
## [31] "New York"     "Niagara"      "Oneida"       "Onondaga"     "Ontario"
## [36] "Orange"       "Orleans"      "Oswego"       "Otsego"       "Putnam"
## [41] "Queens"       "Rensselaer"   "Richmond"     "Rockland"     "stlawrence"
## [46] "Saratoga"     "Schenectady"  "Schoharie"    "Schuyler"     "Seneca"
## [51] "Steuben"      "Suffolk"      "Sullivan"     "Tioga"        "Tompkins"
## [56] "Ulster"       "Warren"       "Washington"   "Wayne"        "Westchester"
## [61] "Wyoming"      "Yates"
```

```
row.names(ny_rwj) <- as.character(tolower(ny_rwj$County))
```

####When comparing the data to the Spaitl Polygons object, we can see the county names are in the same format and join the two together using the `SpatialPolygonsDataFrame` function to create a Spatial Polygons Dataframe. . . . we now have our data joined to our geography.

```
head(row.names(ny_rwj), n=62)
```

```
##  [1] "albany"       "allegany"     "bronx"        "broome"       "cattaraugus"
##  [6] "cayuga"       "chautauqua"   "chemung"      "chenango"     "clinton"
## [11] "columbia"     "cortland"     "delaware"     "dutchess"     "erie"
## [16] "essex"        "franklin"     "fulton"       "genesee"      "greene"
## [21] "hamilton"     "herkimer"     "jefferson"    "kings"        "lewis"
## [26] "livingston"   "madison"      "monroe"       "montgomery"   "nassau"
## [31] "new york"     "niagara"      "oneida"       "onondaga"     "ontario"
## [36] "orange"       "orleans"      "oswego"       "otsego"       "putnam"
## [41] "queens"       "rensselaer"   "richmond"     "rockland"     "stlawrence"
## [46] "saratoga"     "schenectady"  "schoharie"    "schuyler"     "seneca"
## [51] "steuben"      "suffolk"      "sullivan"     "tioga"        "tompkins"
## [56] "ulster"       "warren"       "washington"   "wayne"        "westchester"
## [61] "wyoming"      "yates"
```

```
head(map.IDs, n=62)
```

```
##  [1] "albany"       "allegany"     "bronx"        "broome"       "cattaraugus"
##  [6] "cayuga"       "chautauqua"   "chemung"      "chenango"     "clinton"
## [11] "columbia"     "cortland"     "delaware"     "dutchess"     "erie"
## [16] "essex"        "franklin"     "fulton"       "genesee"      "greene"
## [21] "hamilton"     "herkimer"     "jefferson"    "kings"        "lewis"
## [26] "livingston"   "madison"      "monroe"       "montgomery"   "nassau"
## [31] "new york"    "niagara"     "oneida"      "onondaga"    "ontario"
## [36] "orange"      "orleans"     "oswego"      "otsego"      "putnam"
## [41] "queens"      "rensselaer"  "richmond"    "rockland"    "stlawrence"
## [46] "saratoga"    "schenectady" "schoharie"   "schuyler"    "seneca"
## [51] "steuben"     "suffolk"     "sullivan"    "tioga"       "tompkins"
## [56] "ulster"      "warren"      "washington"  "wayne"       "westchester"
## [61] "wyoming"     "yates"
```

```
ny_rwj_df <- SpatialPolygonsDataFrame(ny_cty_sp,ny_rwj)
summary(ny_rwj_df)
```

```
## Object of class SpatialPolygonsDataFrame
## Coordinates:
##         min       max
## x -79.76718 -71.87756
## y  40.48520  45.01157
## Is projected: TRUE
## proj4string :
## [+init=epsg:2261 +proj=tmerc +lat_0=40 +lon_0=-76.58333333333333
## +k=0.9999375 +x_0=249999.9998983998 +y_0=0 +datum=NAD83
## +units=us-ft +no_defs +ellps=GRS80 +towgs84=0,0,0]
## Data attributes:
##      FIPS              State              County
##  Length:62          Length:62          Length:62
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##    LL.Rank          LL.Quartile          QL.Rank
##  Length:62          Length:62          Length:62
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##  QL.Quartile          HB.Rank          HB.Quartile
##  Length:62          Length:62          Length:62
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##    CC.Rank          CC.Quartile          SE.Rank
##  Length:62          Length:62          Length:62
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##  SE.Quartile          PE.Rank          PE.Quartile
##  Length:62          Length:62          Length:62
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

####Since we are mapping Quanilty of Life and the best county is scored 1, I am going to reverse code the data so that the higher number equals higher quality of life. To do this, we must first create a numeric version of the variable and then substract the variable from 62 to create the reversed range of scores.

```
summary(ny_rwj_df$QL.Rank)
```

```
##    Length     Class      Mode
##        62 character character
```

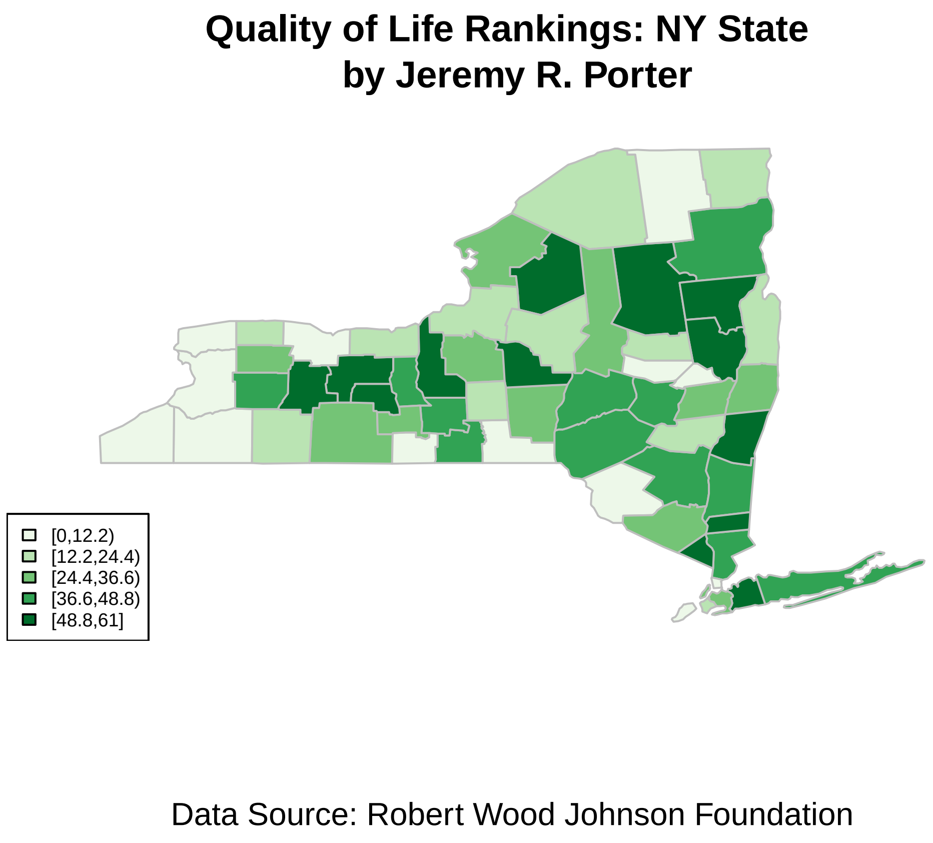
```
ny_rwj_df$QL.Rank <- as.numeric(ny_rwj_df$QL.Rank)
summary(ny_rwj_df$QL.Rank)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   16.25   31.50   31.50   46.75   62.00
```

```
ny_rwj_df$QL.Rank <- 62 - ny_rwj_df$QL.Rank
```

####Now we can use the same approach we took last week to map the data with the internal plotting facilities in our R environment.

```
library(RColorBrewer)
library(classInt)

plotvar <- ny_rwj_df$QL.Rank
nclr <- 5
class <- classIntervals(plotvar, nclr, style = "quantile")
plotclr <- brewer.pal(nclr, "Greens")
colcode <- findColours(class, plotclr, digits = 3)

plot(ny_rwj_df, col = colcode, border = "grey",axes = F)
title(main = "Quality of Life Rankings: NY State \n by Jeremy R. Porter",
      sub = "Data Source: Robert Wood Johnson Foundation")
legend("bottomleft", legend = names(attr(colcode,"table")),
       fill = attr(colcode, "palette"), cex=0.55)
```


Quality of Life Rankings: NY State
by Jeremy R. Porter
[0,12.2)
[12.2,24.4)
[24.4,36.6)
[36.6,48.8)
[48.8,61]
Data Source: Robert Wood Johnson Foundation

####Now that we have a `SpatialPolygonsDataFrame` that is equivalent to what we would think of as `Shapefile`, let's use the `rgdal` package to write the object to a new shapefile in our working file system. The arguments for the `writeOGR` function simply require the identification of a destination `dsn`, a new layer name `layer`, and the types of output file `driver`.

```
require(rgdal)
```

```
## Loading required package: rgdal

## rgdal: version: 1.4-6, (SVN revision 841)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20
##  Path to GDAL shared files: C:/Users/jerem/Documents/R/win-library/3.6/rgdal/gdal
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: C:/Users/jerem/Documents/R/win-library/3.6/rgdal/proj
##  Linking to sp version: 1.3-1
```

```r
writeOGR(ny_rwj_df,
         dsn = "working_directory",
         layer = "RWJ_NY",
         driver = "ESRI Shapefile",
         overwrite_layer = T)
```

####Now let's simply the object in R by reading the data in as a `simple feature` using the `sf` package. Once the data imported, let's explore it with the `names`, `head`, and `plot` functions. Additionally, let's reproject the data to see the difference in visual appearance.

```r
require(sf)
```

```
## Loading required package: sf

## Linking to GEOS 3.6.1, GDAL 2.2.3, PROJ 4.9.3
```

```r
rwj_sf <- st_read(dsn = "working_directory",
                  layer = "RWJ_NY")
```

```
## Reading layer `RWJ_NY' from data source `C:\Prop_val2\stuff\working_directory' using driver `ESRI Sha
## Simple feature collection with 62 features and 15 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: -79.76718 ymin: 40.4852 xmax: -71.87756 ymax: 45.01157
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=40 +lon_0=-76.58333333333333 +k=0.9999375 +x_0=249999.9998983998 
```

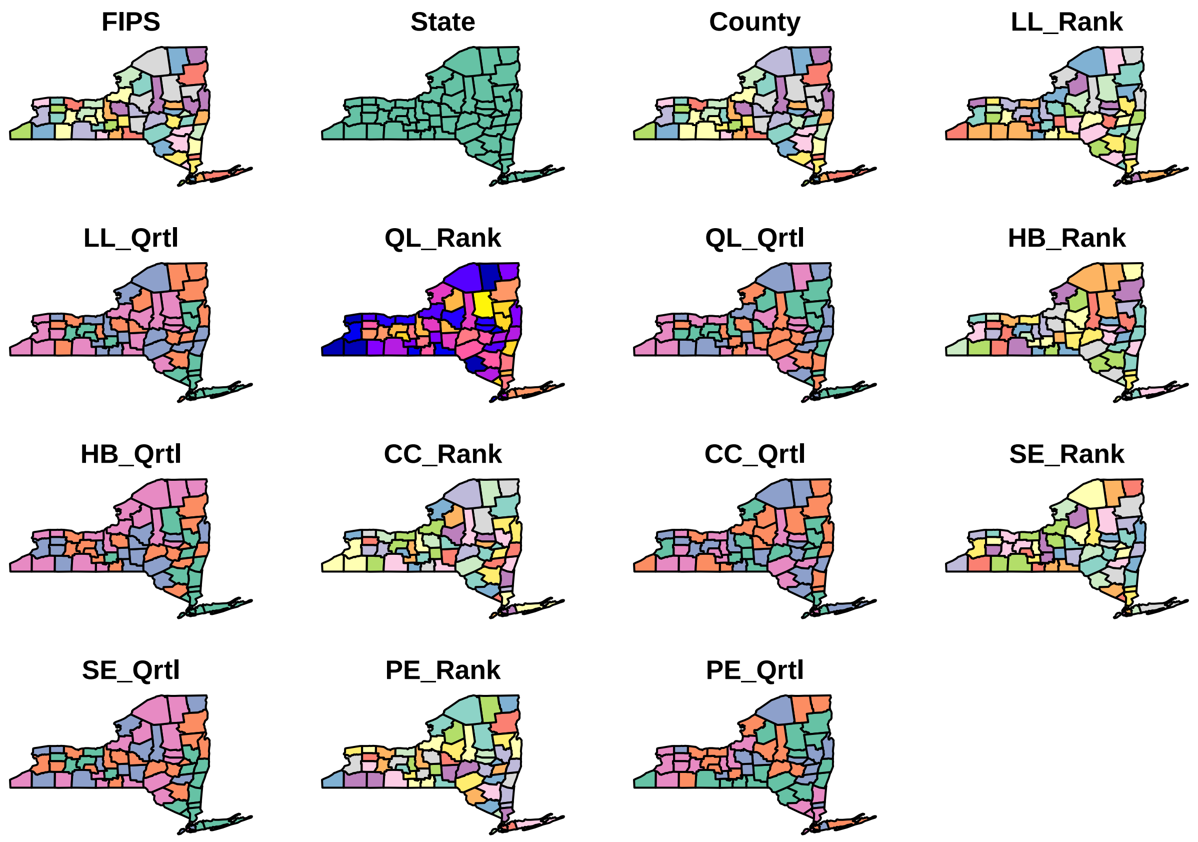
```r
names(rwj_sf)
```

```
##  [1] "FIPS"     "State"    "County"   "LL_Rank"  "LL_Qrtl"  "QL_Rank" 
##  [7] "QL_Qrtl"  "HB_Rank"  "HB_Qrtl"  "CC_Rank"  "CC_Qrtl"  "SE_Rank" 
## [13] "SE_Qrtl"  "PE_Rank"  "PE_Qrtl"  "geometry"
```

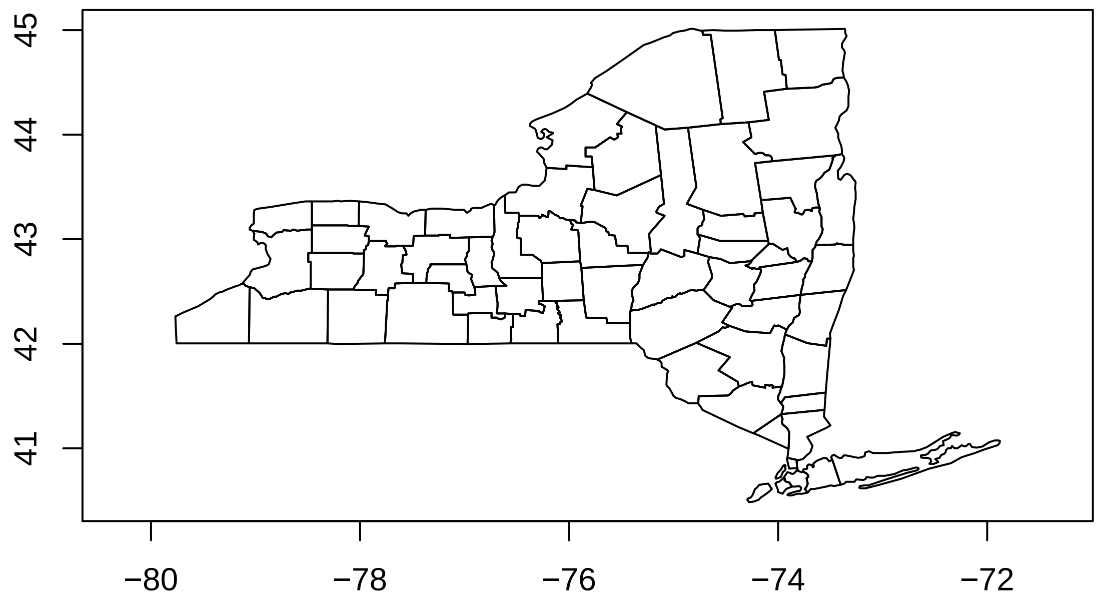
```r
head(rwj_sf)
```

```
## Simple feature collection with 6 features and 15 fields
## geometry type:  POLYGON
## dimension:      XY
## bbox:           xmin: -79.06245 ymin: 40.80605 xmax: -73.67664 ymax: 43.41874
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=40 +lon_0=-76.58333333333333 +k=0.9999375 +x_0=249999.9998983998 
##    FIPS    State      County LL_Rank LL_Qrtl QL_Rank QL_Qrtl HB_Rank
## 1 36001 New York      Albany      21       2      27       3      13
## 2 36003 New York    Allegany      25       2      23       3      45
## 3 36005 New York       Bronx      44       3       0       4      40
## 4 36007 New York      Broome      52       4       9       4      35
## 5 36009 New York Cattaraugus      58       4       2       4      59
## 6 36011 New York      Cayuga      33       3      51       1      55
##   HB_Qrtl CC_Rank CC_Qrtl SE_Rank SE_Qrtl PE_Rank PE_Qrtl
## 1       1       7       1       9       1      36       3
## 2       3      48       4      48       4      29       2
## 3       3      62       4      62       4      59       4
## 4       3      16       1      47       4       8       1
## 5       4      54       4      45       3      50       4
## 6       4      43       3      29       2      47       4
##                            geometry
## 1 POLYGON ((-73.7855 42.46763...
## 2 POLYGON ((-78.20874 41.9978...
## 3 POLYGON ((-73.91728 40.9034...
## 4 POLYGON ((-75.47573 42.0035...
## 5 POLYGON ((-78.92494 42.0035...
## 6 POLYGON ((-76.73051 43.0234...
```

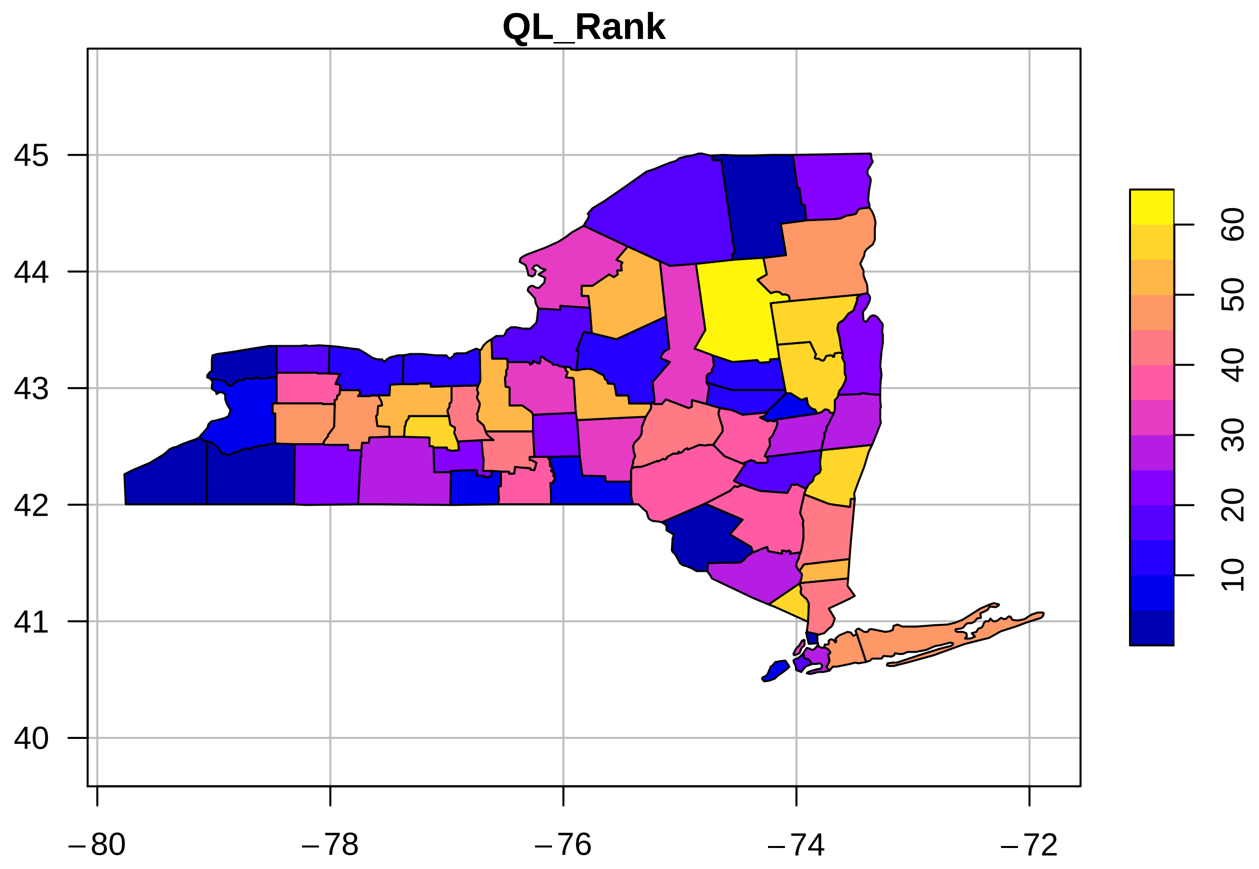
```r
plot(rwj_sf, max.plot = 15)
```



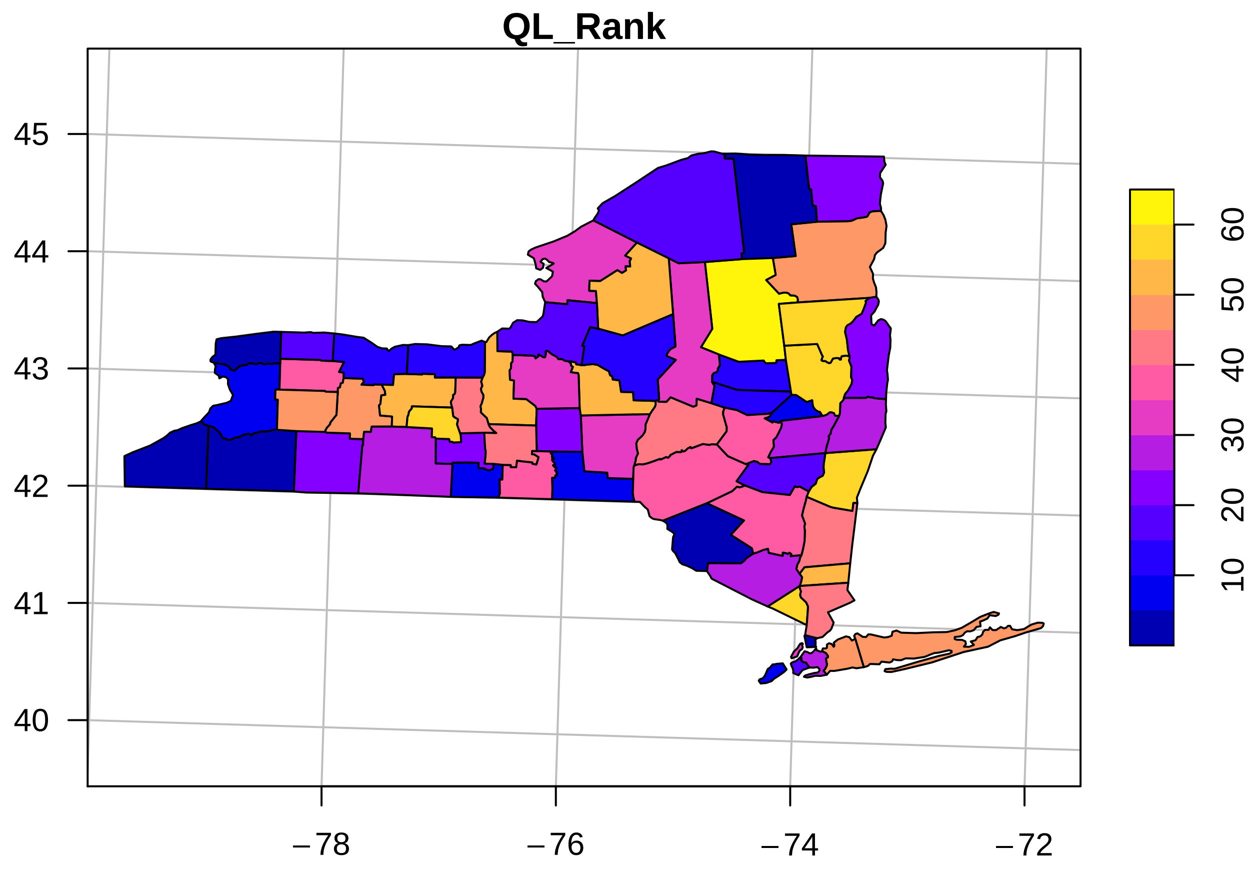
```r
plot(st_geometry(rwj_sf), axes=TRUE)
```



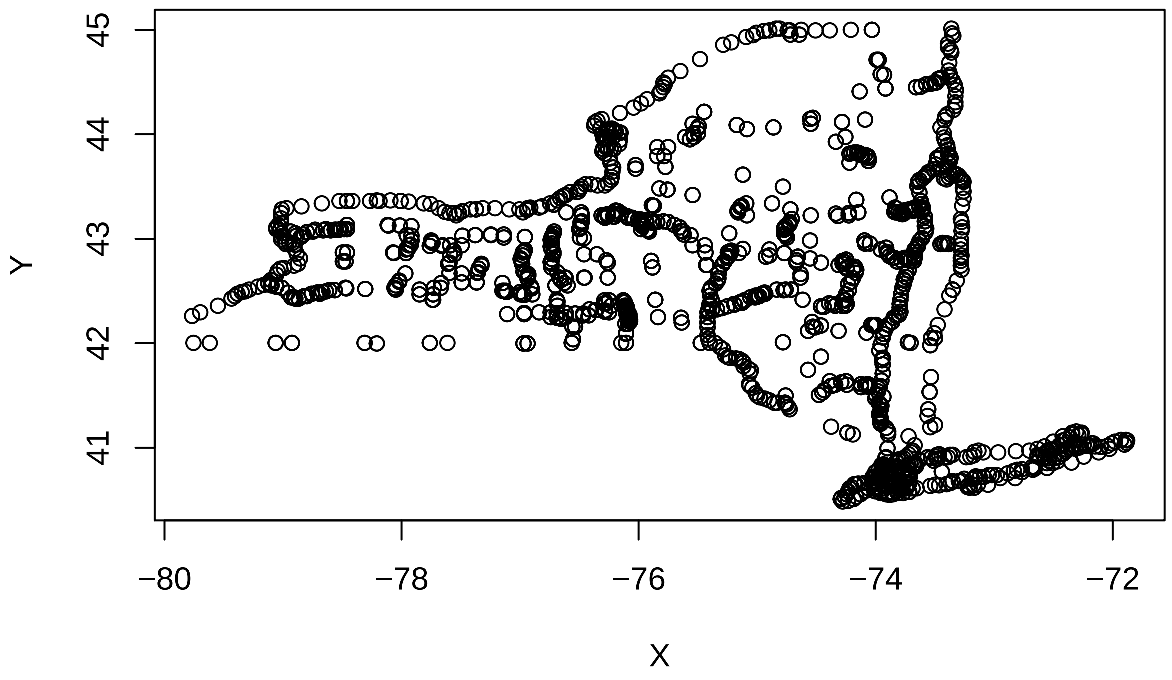
```
plot(rwj_sf[,"QL_Rank"],
     graticule=st_crs(rwj_sf), axes=TRUE, las=1)
```



```r
plot(st_transform(rwj_sf[,"QL_Rank"], 2263),
     graticule=st_crs(rwj_sf),
     axes=TRUE, las=1)
```



```
coords <- st_coordinates(rwj_sf)
plot(coords)
```



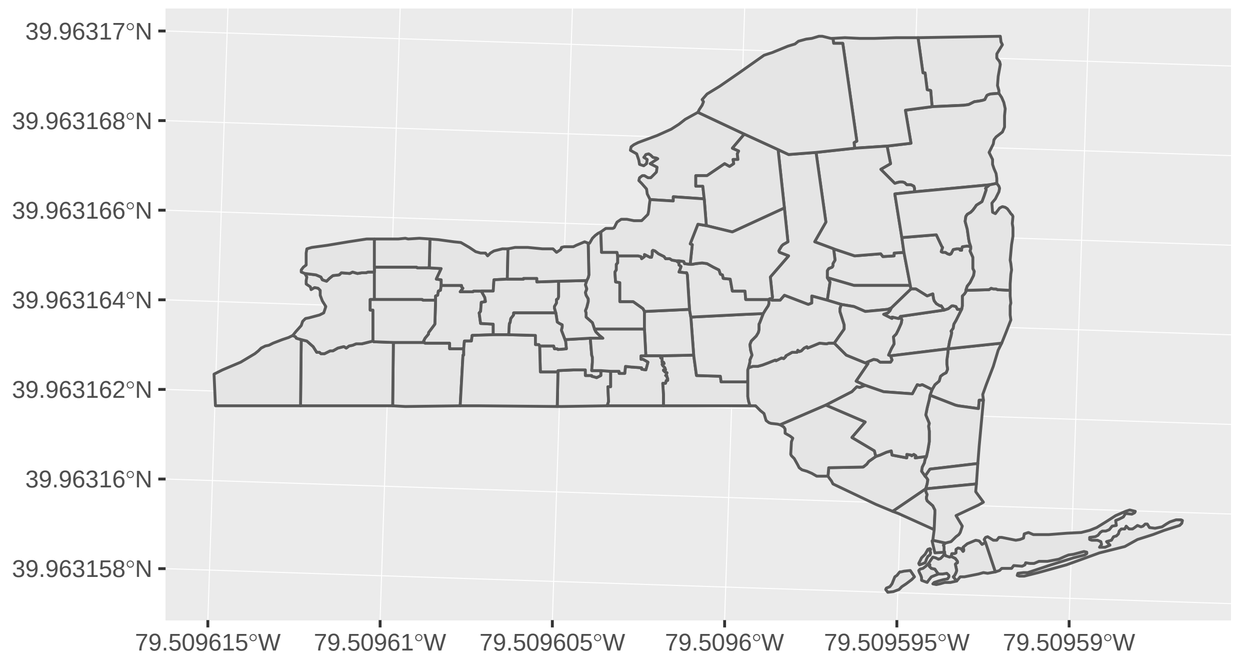
####TO begin, let's install (if needed) and `require` the package `ggplot2` as our mapping tool. To create a simple visualization of our `sf`, let's create a new `ggplot object` called `map 1` where we use the `ggplot` function to map our `data = rjw_sf` and we indicate that the geometry type is a simple feature per the argument `geom_sf()`.

```
#install.packages("ggplot2")
require(ggplot2)
```

```
## Loading required package: ggplot2
```

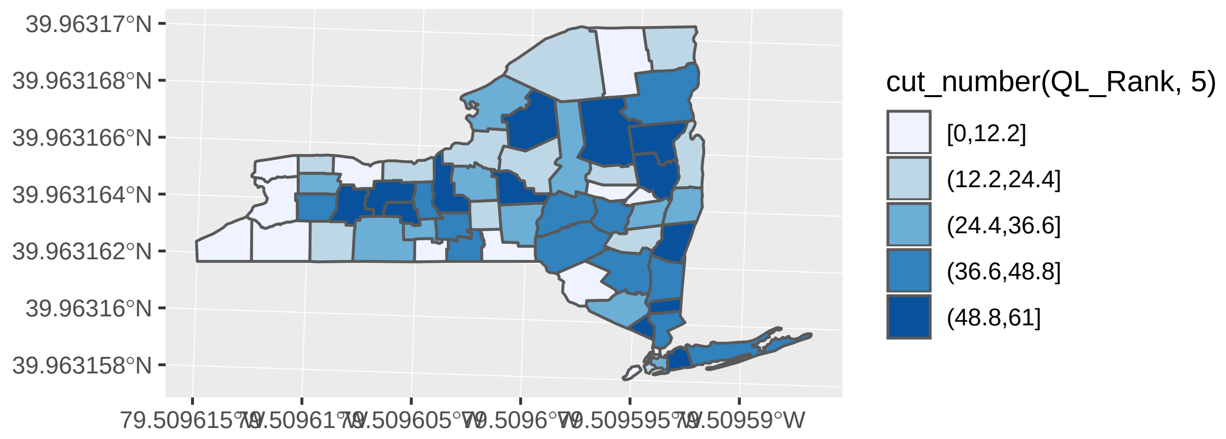
```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
map1 <- ggplot(data = rwj_sf) +
  geom_sf()

map1
```



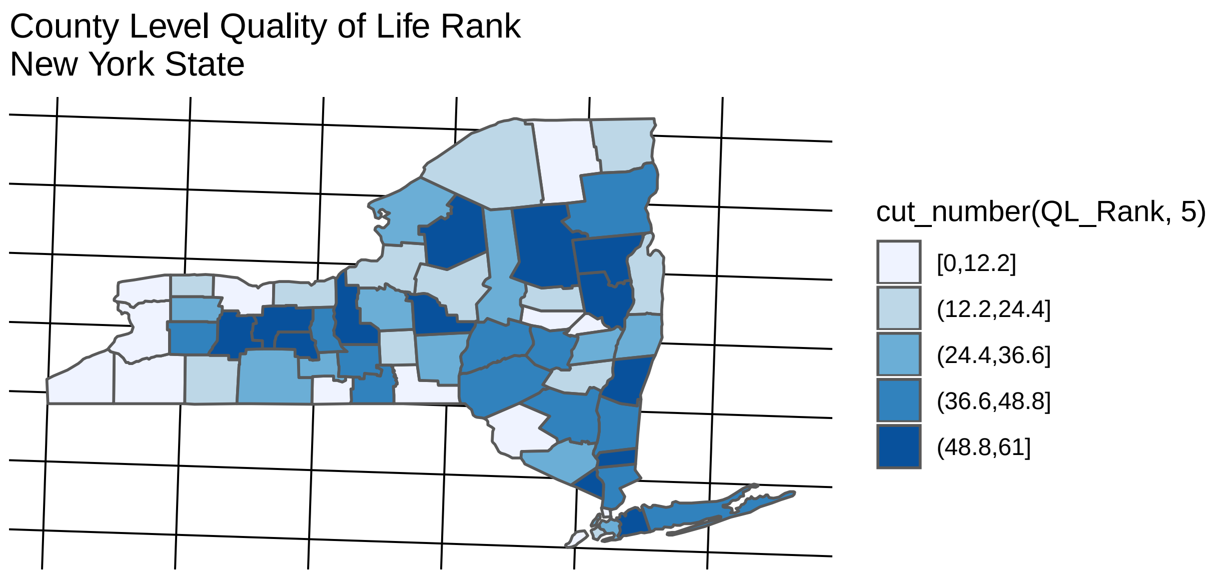
####map 1a builds on the original map by adding a color classification scheme via the aes with the fill argument indicating that the QL.Rank variable should be mapped into 5 classes and visualized with the scale_fill_brewer() default setting.

```
map1a <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer()

map1a
```



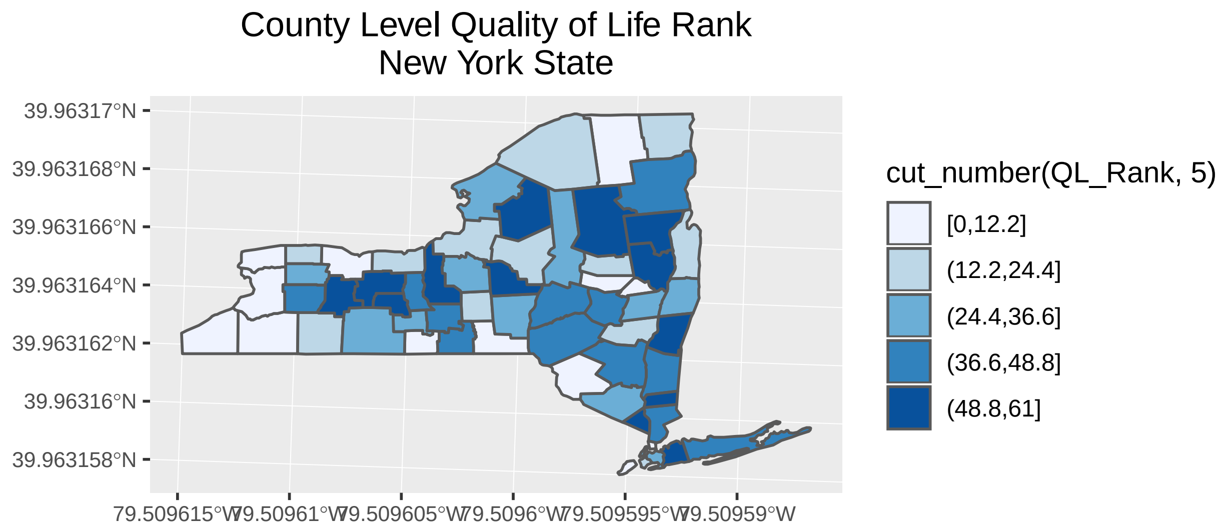
####In the second maps I've added at `theme()` compoent to which I have used to remove the lines, axes, and background using the `element_blank` funtion in the `theme()` area and I added a title using the `ggtitle` function.

```
map2 <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer() +
  ggtitle("County Level Quality of Life Rank\nNew York State") +
  theme(line = element_blank(),
        axis.text=element_blank(),
        axis.title=element_blank(),
        panel.background = element_blank())

map2
```



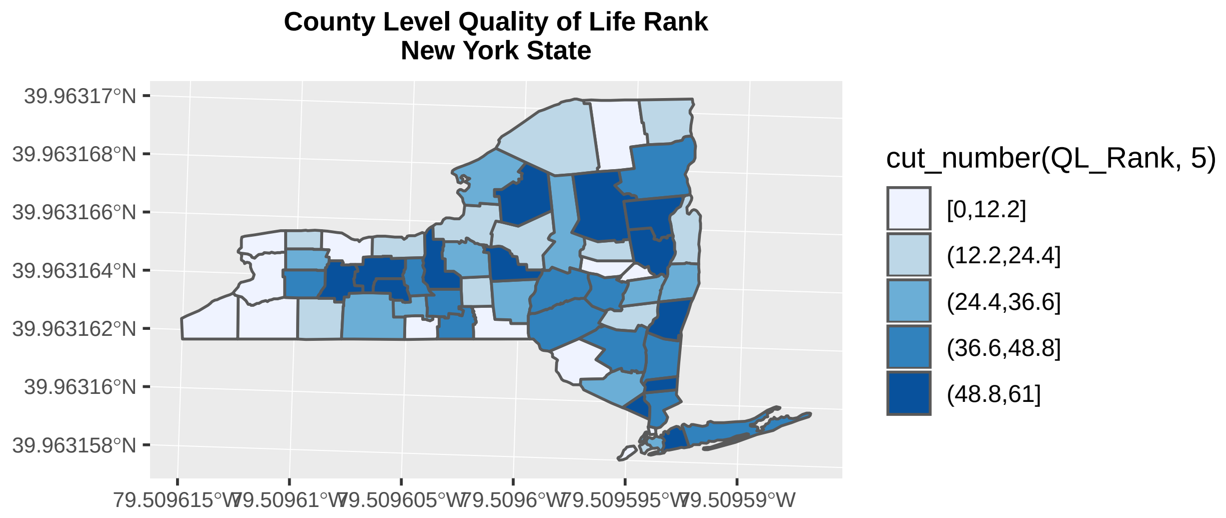
####In the 3rd map I've added back the axes and adjusted the `face` and `size` of the text in the plot.

```
map3 <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer() +
  ggtitle("County Level Quality of Life Rank\nNew York State") +
  theme(axis.text=element_text(size=8),
        axis.title=element_text(size=8,face="bold"),
        plot.title = element_text(hjust = 0.5))

map3
```



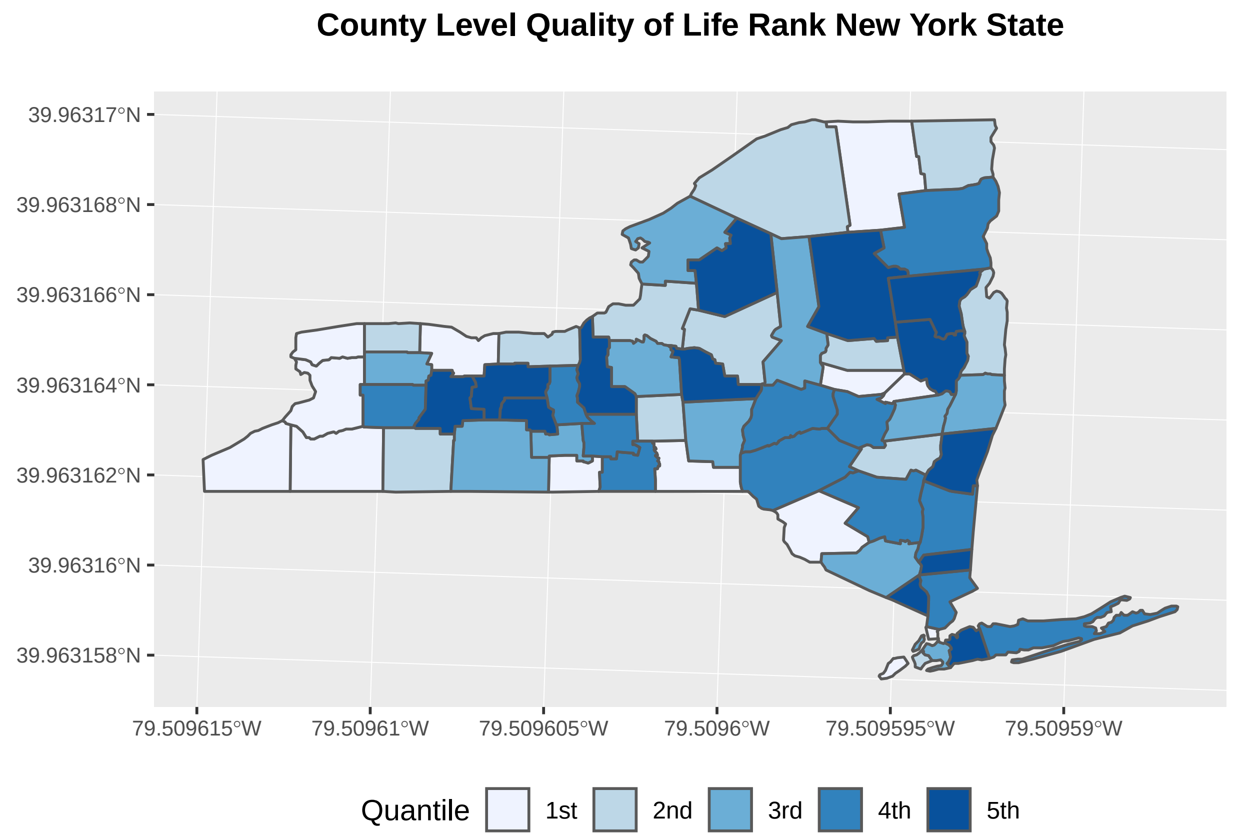
####In the 4th map, I have adjusted the `face`, `size`, and the location (`hjust`) of the title. The `hjust` horizontally adjusts the location of the title along the length of the x-axis, scaled between `0` and `1`. `0.5` is halfway along the x-axis.

```
map4 <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer() +
  ggtitle("County Level Quality of Life Rank\nNew York State") +
  theme(axis.text=element_text(size=8),
        axis.title=element_text(size=8,face="bold"),
        plot.title = element_text(face="bold",size=10,hjust = 0.5))

map4
```



####The 5th map changes the legend symbology using the `scale_fill_brewer` function and name's the legend `Quantile`, set's the `palette` to `Blues`, renames each of the classes using the `labels` function. I also adjusted the legend location to the `bottom` of the map using the `legend.position` function.

```
map5 <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer(name="Quantile", palette="Blues",
                    labels=c("1st",
                             "2nd",
                             "3rd",
                             "4th",
                             "5th")) +
  ggtitle("County Level Quality of Life Rank New York State\n ") +
  theme(axis.text=element_text(size=8),
        axis.title=element_text(size=8),
        plot.title = element_text(face="bold",size=12,hjust = 0.5),
        legend.position="bottom")

map5
```



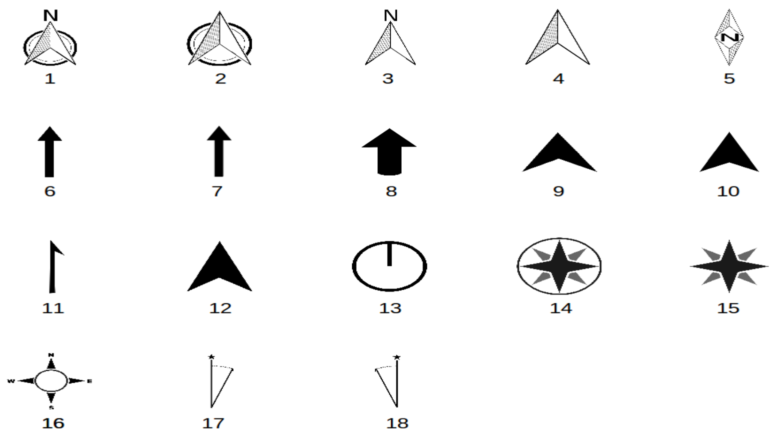
####To add map elements, sometimes we need additional packages. One that allows for the placement of common cartographic elements is the `ggsn` package. In particular, we are going to be interested in extracting a north arrow form the package. Once the pacakge is loaded, we can use the `northSymbols()` function to see north arrow options.

```
#install.packages("ggsn")
require(ggsn)
```

```
## Loading required package: ggsn
```

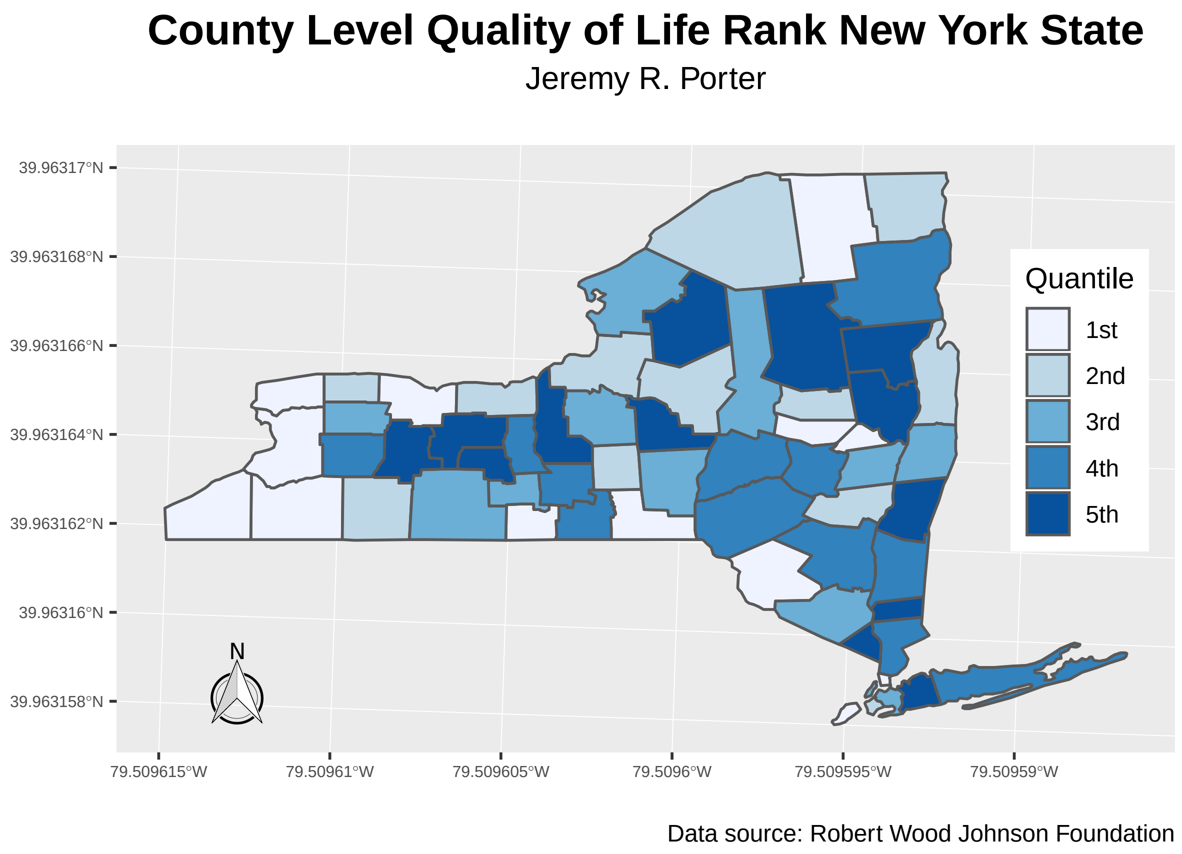
```
## Loading required package: grid
```

```
northSymbols()
```



####Finally, the 6th map adjusts font sizes for titles, axis labels, and legend information while alos relocating the legend to the righthand side of the map with the `legend.position` function which locates the legend `91%` of the length of the `x-axis` and `58%` of the way up the `y-axis`. The `ggsn` package also allows us to add a north arrow with `north` function and a scalebar with the `scalebar` function.

```
map6 <- ggplot(data = rwj_sf) +
  geom_sf() +
  aes(fill=cut_number(QL_Rank, 5)) +
  scale_fill_brewer(name="Quantile", palette="Blues",
                    labels=c("1st",
                             "2nd",
                             "3rd",
                             "4th",
                             "5th")) +
  labs(title = "County Level Quality of Life Rank New York State",
       subtitle = "Jeremy R. Porter\n",
       caption = "\nData source: Robert Wood Johnson Foundation") +
  theme(axis.text=element_text(size=6),
        axis.title=element_text(size=6),
        plot.title = element_text(face="bold",size=16,hjust = 0.5),
        plot.subtitle = element_text(size=12,hjust = 0.5),
        plot.caption = element_text(),
        legend.position=c(0.91,0.58)) +
  north(rwj_sf, scale = 0.15, symbol=1, location="bottomleft")


map6
```



####You can export this final map using the `export` option in the `plots` area of `r-studio`. For your homework, create a map of the Health Behaviors Ranking `HB.Rank` for the state of `New Jersey` export that NJ-Health Behaviors map and submit it as your lab deliverable.