CountyStat PDF Template

Tue June 15, 2021 11:49 AM

Contents

Getting Started

For many of the basics in RMarkdown the RStudio introduction is very worth while. Almost every R visualization output is compatible with it.

For this report standard if other images need to be replaced with others that is fine.

Common Standards

Database Connections

```
require(DBI)

wh_host <- Sys.getenv('WH_HOST')
wh_db <- Sys.getenv('WH_DB')
wh_user <- Sys.getenv('WH_USER')
wh_pass <- Sys.getenv('WH_PASS')

# Connect to the CountyStat DataWarehouse
wh_con <- dbConnect(odbc::odbc(),
                    driver = "{ODBC Driver 17 for SQL Server}",
                    server = wh_host,
                    database = wh_db,
                    UID = wh_user,
                    pwd = wh_pass,
                    Trusted_Connection= "yes")
```

Always keep your credentials for the DataWarehouse in an `.env` file. An example is in this repo, but in normal circumstances this file should be in the `.gitignore` file.

To use DBI check out their documentation, but for the most part you will be using either the `dbGetQuery()` or `dbReadTable()` function.

Reading Flat Files

For flat files, `readr` will do well for most CSV's and other basic file formats, `readxl` is best for excel documents, `jsonlite` is ideal for JSON documents, and `xml2` will do if you have the unfortunate circumstance of dealing with that format.

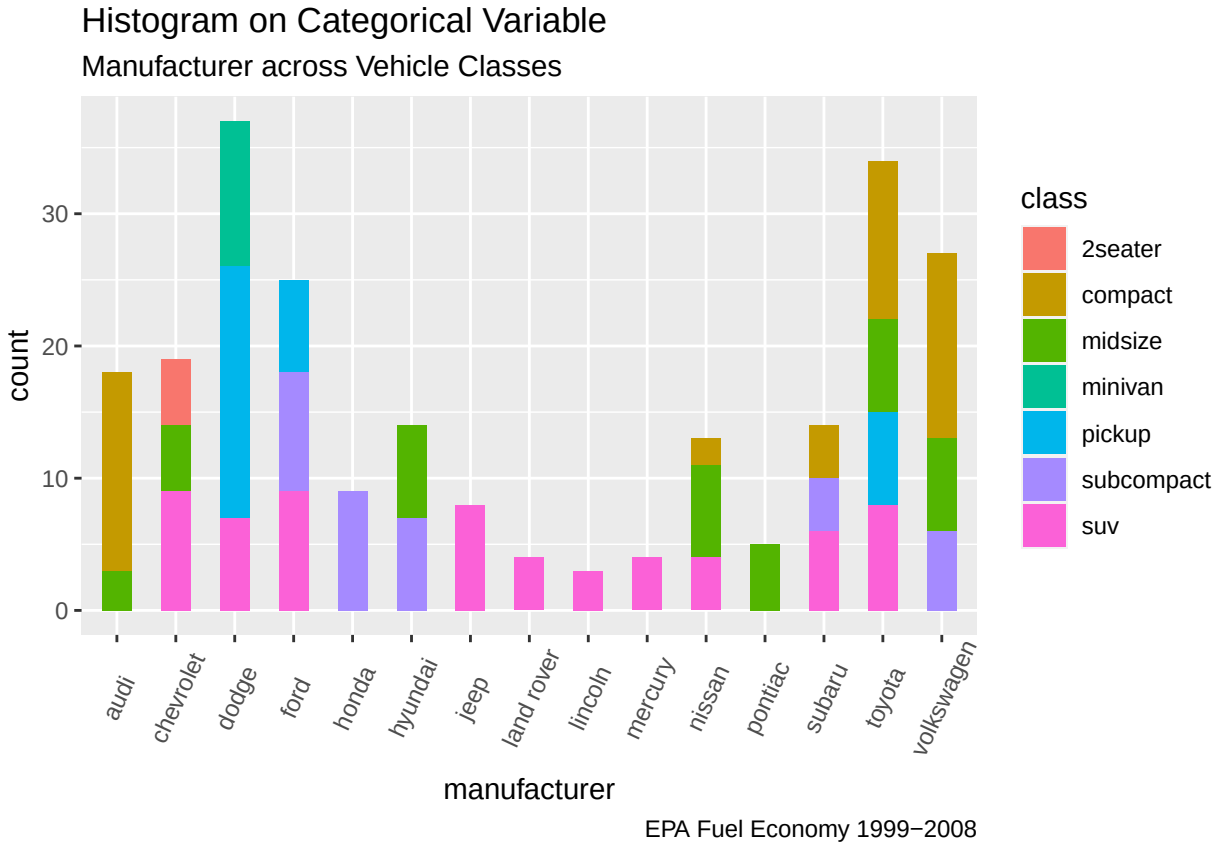
```
require(readr)
require(readxl)
require(jsonlite)
require(xml2)
```

Data Visualizations

The easiest package for quickly creating charts and graphs in R is ggplot2. Once of the best resources for ggplot2 visualizations is the Topic 50 ggplot2 Visualizations from Selva Prabhakaran. This includes full code and examples using you can access directly from R. This site also has a good introduction to ggplot2 if you find the one from RStudio insufficient.

```
require(ggplot2)

g <- ggplot(mpg, aes(manufacturer)) + geom_bar(aes(fill=class), width = 0.5) +
```

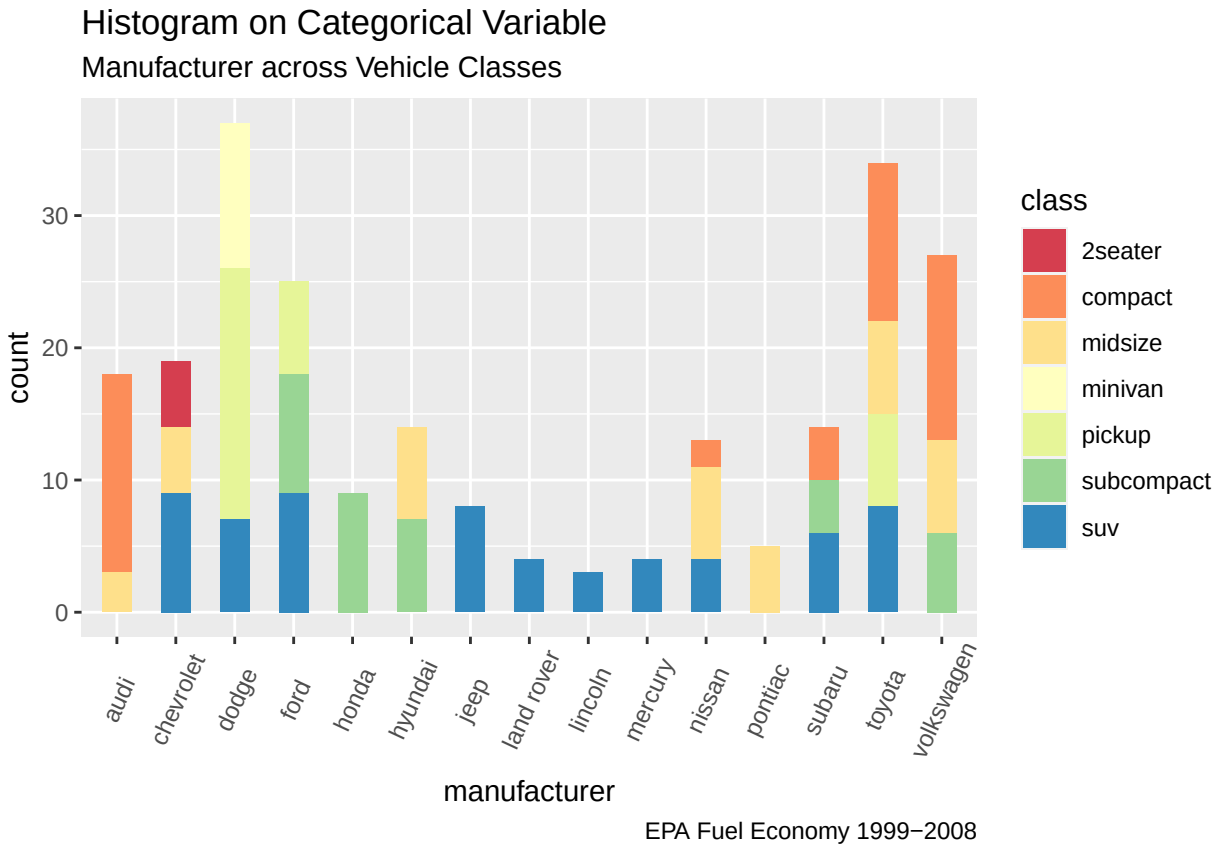
```
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Histogram on Categorical Variable",
       subtitle="Manufacturer across Vehicle Classes",
       caption = "EPA Fuel Economy 1999-2008")

g
```



Colors

A great resource for color palettes is colorbrewer, which is easy to implement in ggplot2.

```
g + scale_fill_brewer(palette = "Spectral")
```



Tables

The best functions for tables are `DT()` and `kable()`. The best option will depend on the number of rows and situation you find yourself in. Both of these functions allow for a wide amount of customization and formating. CountyStat has a standard table format to start with, however Analysts should feel free to check out the documentation for both packages especially when looking to format the rows to highlight certain values or increase readability, or improve other functions.

kable w/ kableExtra

For PDF's kableExtra package to format your `kable()` tables are your best bet. Below is a standard CountyStat format is available below, but Analysts should feel free to use the wide variety of features the `kableExtra` library provides to improve readability and convey a message.

```
require(kableExtra)

iris %>%
  sample_n(15) %>% # Select 15 random rows
  kbl(caption = "Fisher's or Anderson's") %>%
  kable_styling(font_size = 12) %>%
  row_spec(0, bold = T, background = "#008080", color = "white")
```

Table 1: Fisher's or Anderson's

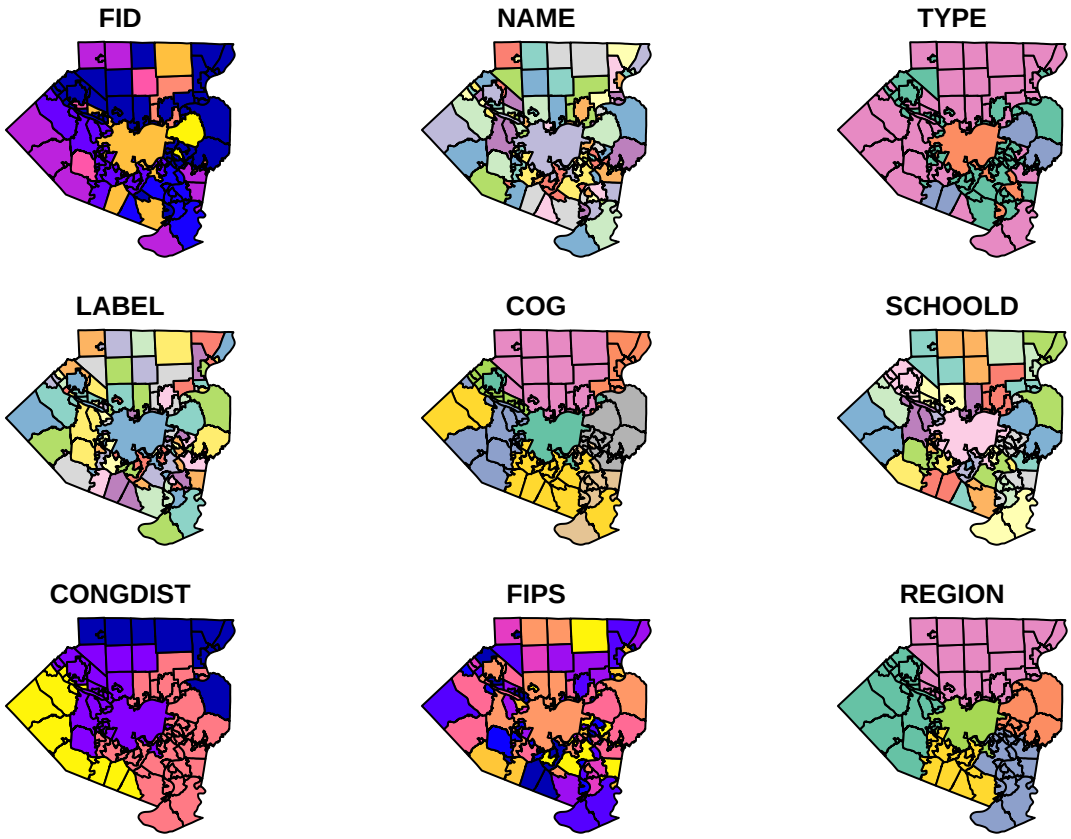
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.6	3.0	4.5	1.5	versicolor
5.0	3.3	1.4	0.2	setosa
6.4	3.2	5.3	2.3	virginica
5.2	3.5	1.5	0.2	setosa
4.7	3.2	1.3	0.2	setosa
6.5	2.8	4.6	1.5	versicolor
6.7	3.1	4.4	1.4	versicolor
4.7	3.2	1.6	0.2	setosa
5.6	2.9	3.6	1.3	versicolor
5.6	3.0	4.1	1.3	versicolor
5.1	3.3	1.7	0.5	setosa
5.8	2.7	5.1	1.9	virginica
5.1	2.5	3.0	1.1	versicolor
6.9	3.1	5.4	2.1	virginica
5.2	4.1	1.5	0.1	setosa

Maps and Spatial Data

Reading Data

The easiest method for reading data and mapping in either `ggplot2`, `ggmap` or `leaflet` is the sf package. When looking for a spatial dataset, the first option is always what is available from the County ESRI team on their GIS Open Data site. If you can't find what you are looking for, and it is something you think the County should have access to email their team at GISHelp@AlleghenyCounty.US.

```
require(sf)

munis <- read_sf('https://opendata.arcgis.com/datasets/9de0e9c07af04e638dbc9cb9070962c2_0.geojson')

plot(munis)
```



Census Data

The easiest way to use the Census API in R is with the tidycensus package. First you have to sign-up for an api key, but its a free and easy process.

```
require(tidycensus)

census_api_key(Sys.getenv("census"))

v19 <- load_variables(2019, "acs5", cache = TRUE)

alco_muni_pop <- get_acs("county subdivision", state = 'PA', county = 'Allegheny', year = 2019, variables =
```

```
##   |                                                                              |
```

```
require(ggmap)
require(ggthemes)

bbox <- st_bbox(munis)
bbox_trans <- c(left = bbox[[1]], bottom = bbox[[2]], right = bbox[[3]], top = bbox[[4]])

get_stamenmap(bbox_trans, maptype = "toner-lite") %>%
```

```
ggmap() +
theme_map() +
geom_sf(data = alco_muni_pop, aes(fill = estimate), inherit.aes = FALSE) +
scale_fill_gradient(low = '#efedf5', high = '#3f007d')
```