A Comprehensive Guide to the Grammar of Graphics for Effective Visualization of Multi-dimensional Data

Learn effective strategies for leveraging a layered Grammar of Graphics framework for effective data visualization

Dipanjan (DJ) Sarkar · Following

Published in Towards Data Science

11 min read · Sep 12, 2018

Listen Share More

Introduction

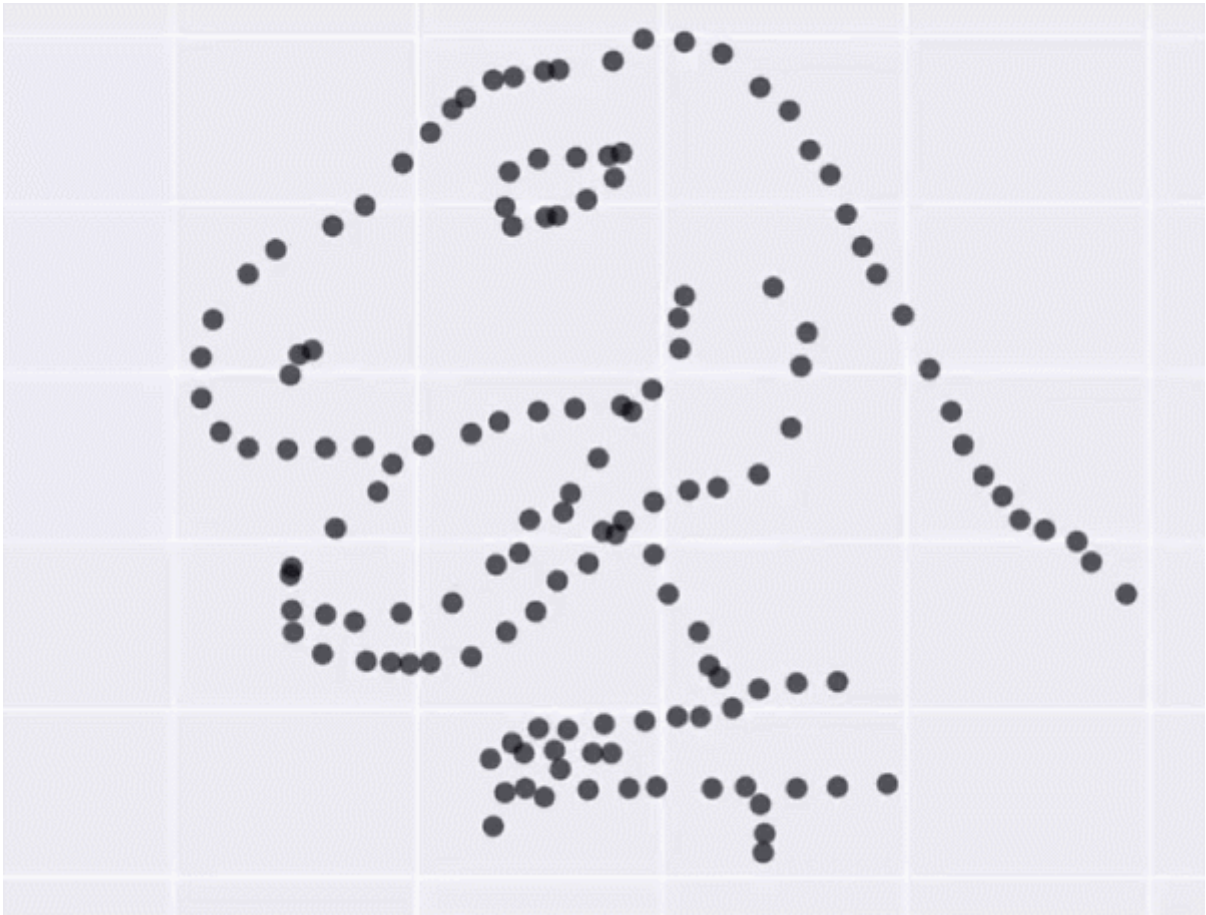
Visualizing multi-dimensional data is an art as well as a science. Due to the limitations of our two-dimensional (2-D) rendering devices, building effective visualizations on more than two data dimensions (attributes or features) becomes

challenging as the number of dimensions start increasing. We have extensively covered some strategies for effective multi-dimensional data visualization in one of my previous articles, *"The Art of Effective Visualization of Multi-dimensional Data"* with hands-on examples. In this article, we will cover the layered framework which we leveraged to build these visualizations, called ***'The Grammar of Graphics'***. We will also explore essential concepts behind the layered Grammar of Graphics framework and discuss how we can use each specific layered component to build effective visualization on multi-dimensional data. Examples will be shown in Python, however, if you are interested you can replicate the same in R with ease.

Motivation

Data visualization and storytelling has always been one of the most important phases of any data science pipeline involving extracting meaningful insights from data, regardless of the complexity of the data or the project. Take a simple example of *'The Datasaurus Dozen'* — twelve different datasets depicted in the following figure.

> *Can you guess what is common among these very different looking sets of data?*

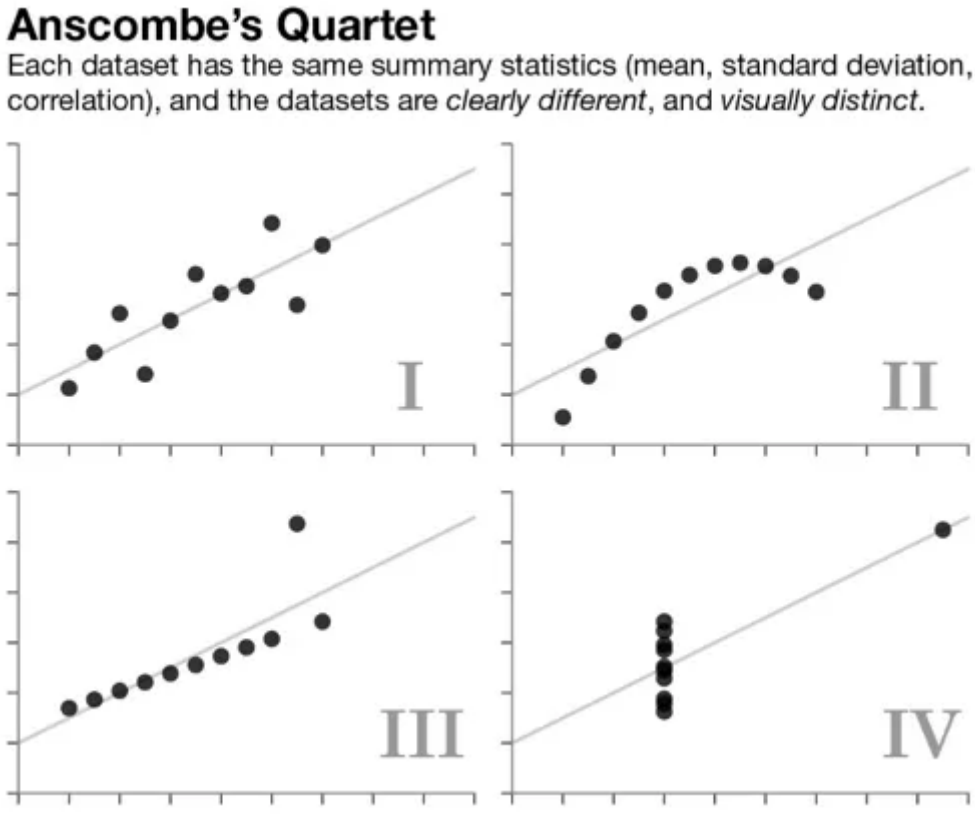


The Datasaurus Dozen — What is common among these diverse datasets?

> *Answer: Summary statistics for all the datasets are exactly the same!*

```
X Mean: 54.26
Y Mean: 47.83
X SD  : 16.76
Y SD  : 26.93
Corr. : -0.06
```

This is a fun variant of the well known Anscombe's quartet, which many of you might be very familiar with, as depicted in the following figure.

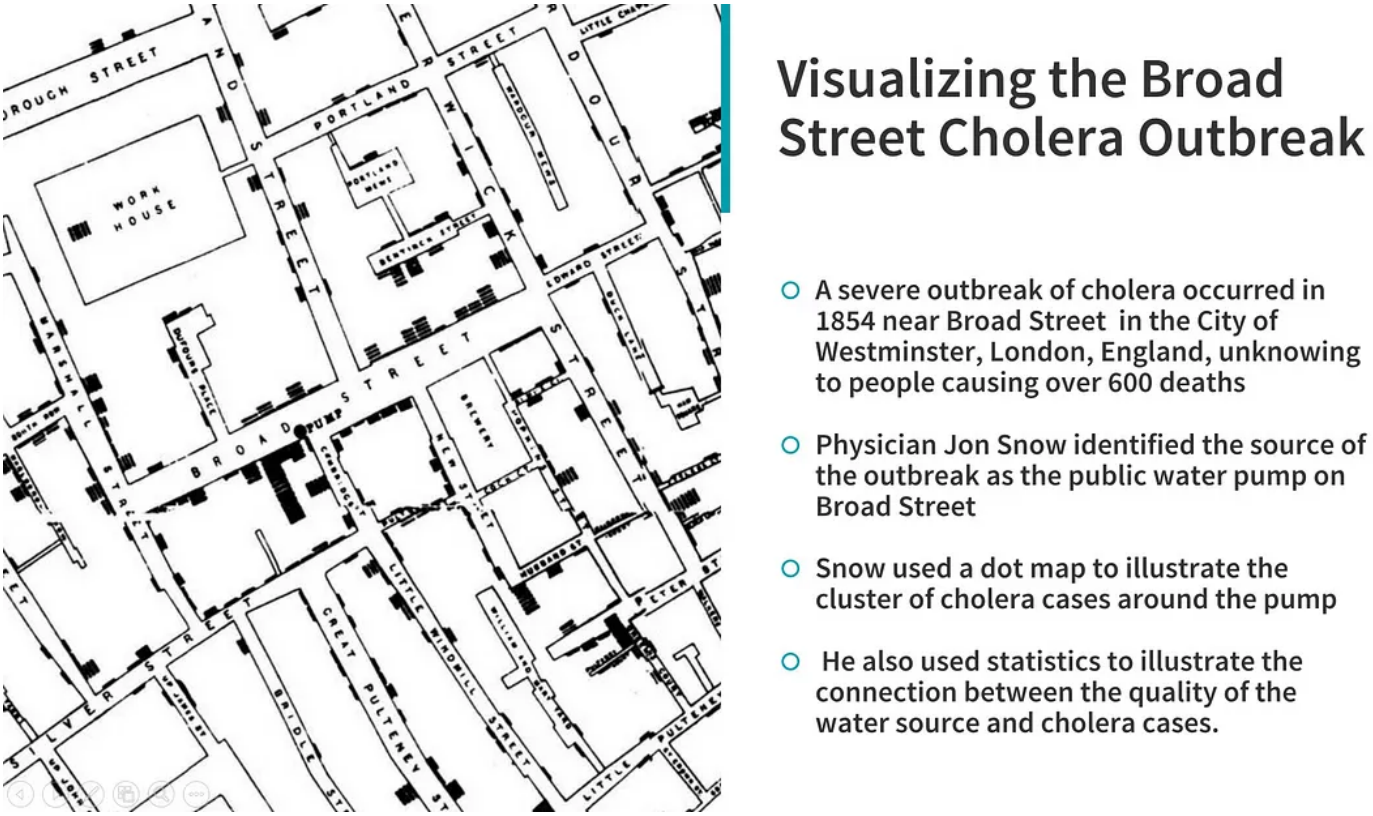


Anscombe's quartet — Different datasets with same summary statistics

The key takeaway from these demonstrations would be, *"Do not trust your data blindly, and start modeling on your data"*. Summary statistics can always be deceptive. Always visualize and understand your data attributes before moving on

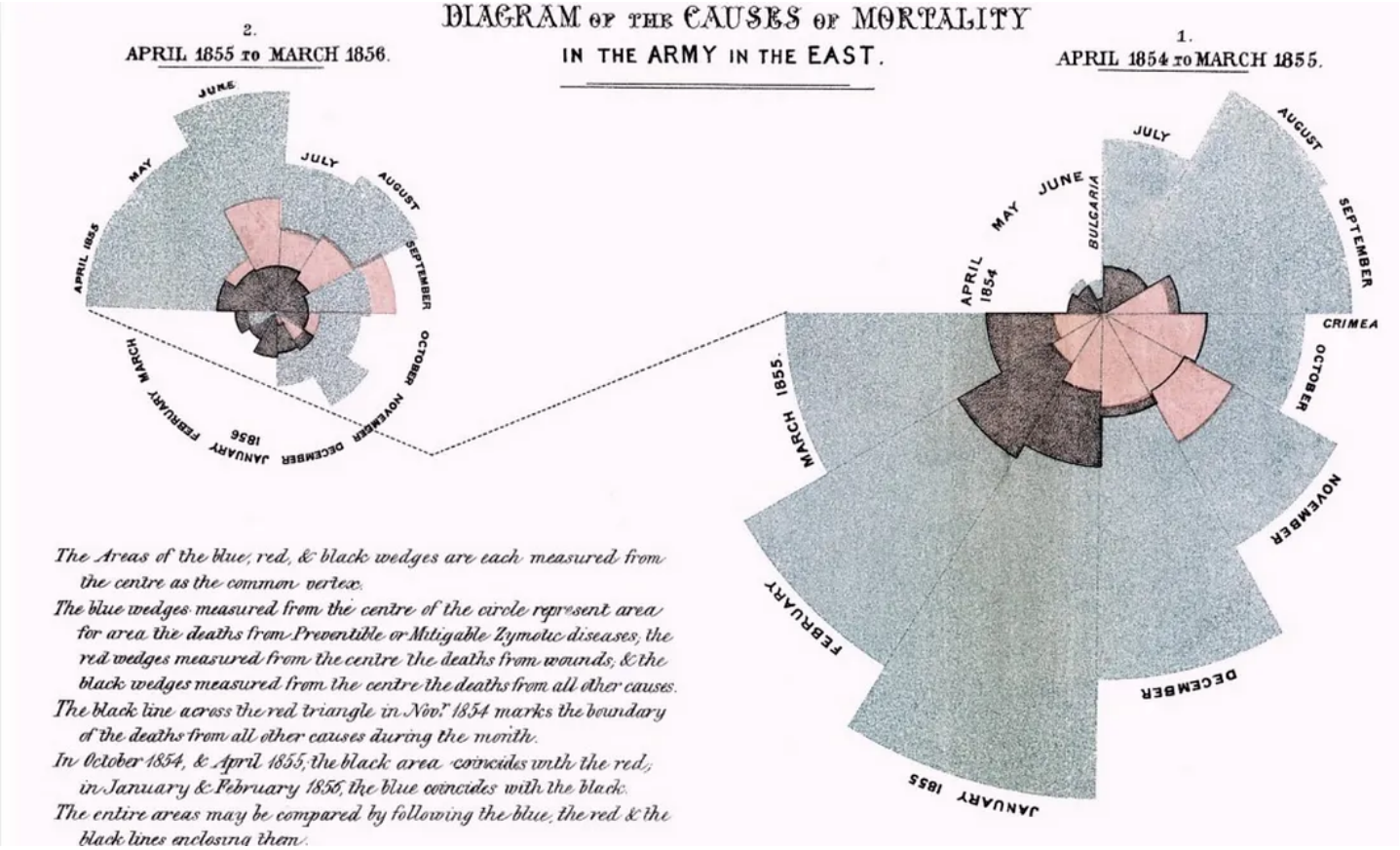
to feature engineering and building statistical, machine learning and deep learning models.

Another very important source of motivation, particularly for effective data visualization, can be derived from some excellent case-studies dating several centuries back when we didn't even have computers let alone Python or R! The first one is John Snow's famous visualization depicting the Broad Street Cholera Outbreak in London, England in 1854!



Visualizing the Broad Street Cholera outbreak which helped find the root cause of the disease outbreak!

You can see how a simple hand-drawn visualization helped find the root cause of the cholera outbreak in Broad Street way back in the 1850s. Another interesting visualization was built by Florence Nightingale, the mother of modern nursing practice, who had a deep-seated interest in nursing and statistics.

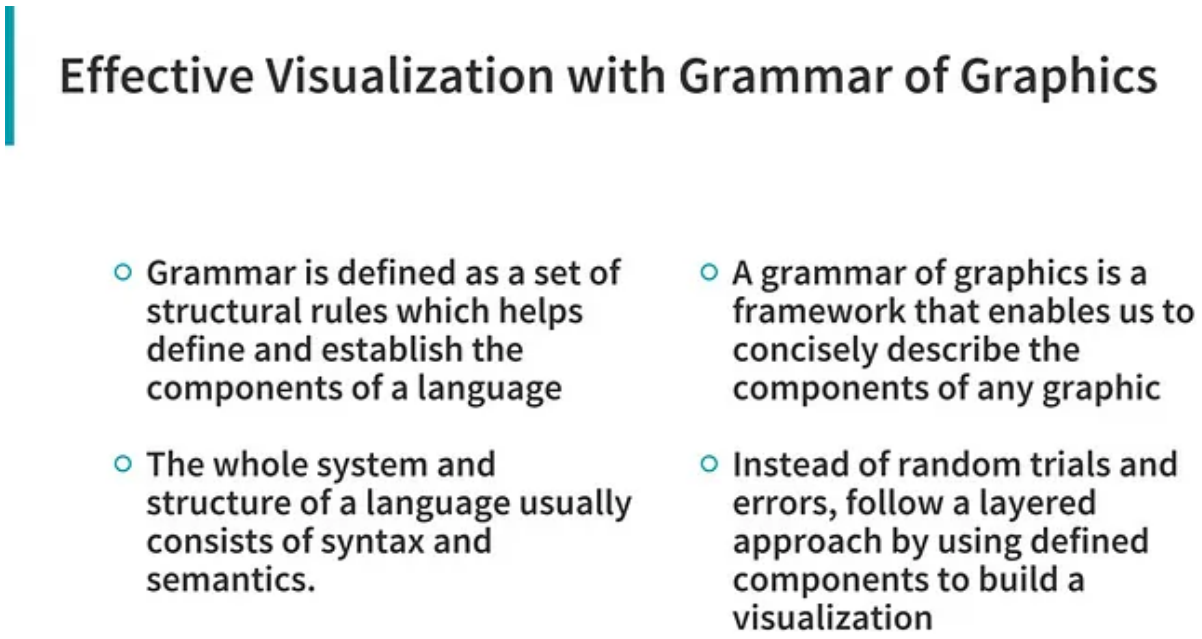


Causes of Mortality in the Army of the East — Florence Nightingale

The above figure depicts a polar area diagram depicting causes of mortality (death) in the army in the 1850s. We can see the visualization is definitely not simplistic, yet it conveys the right insights — clearly showing the proportion of soldiers who died due to diseases which were preventable, based on wounds or other causes. This should serve as enough motivation for effective data visualization!

Understanding the Grammar of Graphics

To understand the Grammar of Graphics, we would need to understand what do we mean by Grammar. The following figure summarizes both these aspects briefly.



Basically, a grammar of graphics is a framework which follows a layered approach to describe and construct visualizations or graphics in a structured manner. A visualization involving multi-dimensional data often has multiple components or aspects, and leveraging this layered grammar of graphics helps us describe and understand each component involved in visualization — in terms of data, aesthetics, scale, objects and so on.

The original grammar of graphics framework was proposed by Leland Wilkinson, which covers all major aspects pertaining to effective data visualization in detail. I would definitely recommend interested readers to check out the book on it, whenever they get a chance!

The Grammar of Graphics | Leland Wilkinson | Springer

Preface to First Edition Before writing the graphics for SYSTAT in the 1980's, I began by teaching a seminar in...

www.springer.com

We will, however, be using a variant of this — known as the layered grammar of graphics framework, which was proposed by Hadley Wickham, reputed Data Scientist and the creator of the famous R visualization package `ggplot2` . Readers should check out his paper titled, *'A layered grammar of graphics'* which covers his

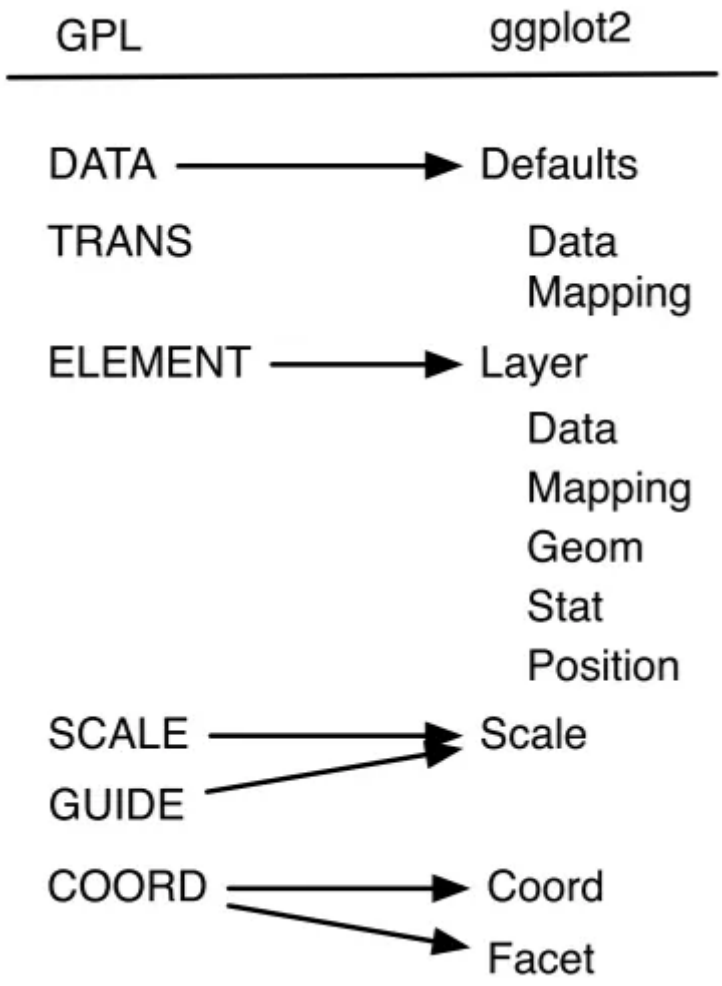
proposed layered grammar of graphics in detail and also talks about his open-source implementation framework `ggplot2` which was built for the R programming language

A layered grammar of graphics

A grammar of graphics is a tool that enables us to concisely describe the components of a graphic. Such a grammar...

vita.had.co.nz

Hadley's layered grammar of graphics uses several layered components to describe any graphic or visualization. Most notably, it has some variations from the original grammar of graphics proposed by Wilkinson as depicted in the following figure.

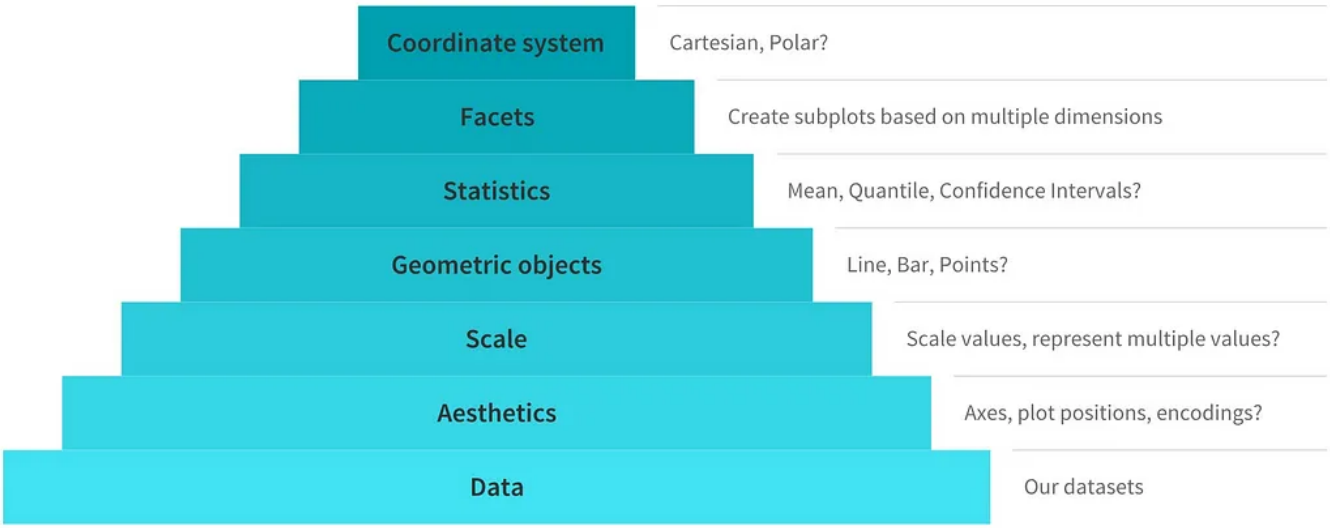


Mapping between components of Wilkinson's grammar (left) and the layered grammar (right)

You can see there are various components in the grammar which can be used to build and describe a visualization. I have identified *seven* such major components which usually help me build effective visualizations on multi-dimensional data. The

following figure illustrates it with some details about each specific component in the grammar.

Major Components of the Grammar of Graphics



Major components of the Grammar of Graphics

We illustrate the same using a pyramid architecture to show an inherent layered hierarchy of components. Typically, to build or describe any visualization with one or more dimensions, we can use the components as follows.

1. **Data:** Always start with the data, identify the dimensions you want to visualize.
2. **Aesthetics:** Confirm the axes based on the data dimensions, positions of various data points in the plot. Also check if any form of encoding is needed including size, shape, color and so on which are useful for plotting multiple data dimensions.
3. **Scale:** Do we need to scale the potential values, use a specific scale to represent multiple values or a range?
4. **Geometric objects:** These are popularly known as ‘geoms’. This would cover the way we would depict the data points on the visualization. Should it be points, bars, lines and so on?

5. **Statistics:** Do we need to show some statistical measures in the visualization like measures of central tendency, spread, confidence intervals?
6. **Facets:** Do we need to create subplots based on specific data dimensions?
7. **Coordinate system:** What kind of a coordinate system should the visualization be based on — should it be cartesian or polar?

We will now be looking at how to leverage this layered framework to build effective data visualizations for multi-dimensional data with some hands-on examples.

Grammar of Graphics in Action

We will now apply the concept we learnt in the previous section on some actual data. We will be building all our visualizations in Python, but I would also recommend people to check out the `ggplot2` package in R, which is one of the most inspiring frameworks till now for building nice and clean publication quality visualizations. We do have a `ggplot` framework in Python which has been built by Yhat, Inc. However, if you check the repository, no commits or updates have been pushed for the last two years, and unfortunately, this causes some errors especially due to backward incompatibility issues with newer versions of `pandas`. Hence, to emulate the true layered grammar of graphics syntax in Python, we will use another interesting framework called `plotnine`.

A Grammar of Graphics for Python - plotnine 0.4.0 documentation

plotnine is an implementation of a grammar of graphics in Python, it is based on ggplot2. The grammar allows users to...

plotnine.readthedocs.io

Plotnine is an open-source Python implementation for a layered grammar of graphics framework which is based on `ggplot2`. Thus, using the previously specified components in this layered grammar, we can build effective visualizations.

Start with the data

We always start by loading up and looking at the dataset we want to analyze and visualize. We will use the famous `mtcars` dataset available as one of the pre-loaded datasets in `plotnine`.

```
from plotnine import *
from plotnine.data import mtcars

mtcars.head()
```

gog_data_viz1.py hosted with ❤ by GitHub view raw

	name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

A data frame with 32 observations on 11 (numeric) variables.

[, 1] mpg Miles/(US) gallon
[, 2] cyl Number of cylinders
[, 3] disp Displacement (cu.in.)
[, 4] hp Gross horsepower
[, 5] drat Rear axle ratio
[, 6] wt Weight (1000 lbs)
[, 7] qsec 1/4 mile time
[, 8] vs Engine (0 = V-shaped, 1 = straight)
[, 9] am Transmission (0 = automatic, 1 = manual)
[,10] gear Number of forward gears
[,11] carb Number of carburetors

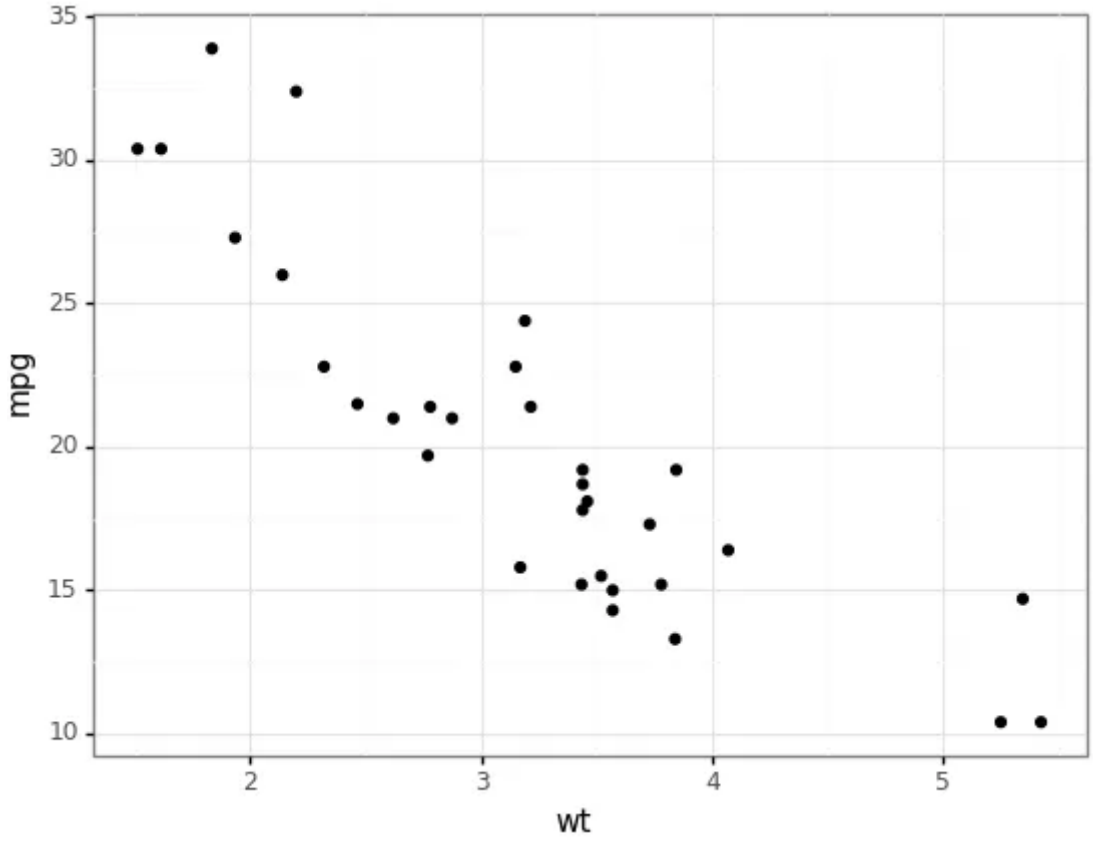
The mtcars dataset consists of data that was extracted from the 1974 *Motor Trend* US magazine, and depicts fuel consumption and 10 other attributes of automobile design and performance for 32 automobiles (1973–74 models). The details of each attribute are depicted in the figure above. Let's build some visualizations now.

Visualize two-dimensions (2-D)

We can now visualize data up to two dimensions using some of the components from our layered grammar of graphics framework including data, scale, aesthetics and geoms. We choose a dot or scatter plot in this case for our geometric object to represent each data point.

```
(ggplot(mtcars,
        aes('wt', 'mpg'))
        + geom_point()
        + theme_bw())
```

gog_data_viz2.py hosted with ❤ by GitHub view raw



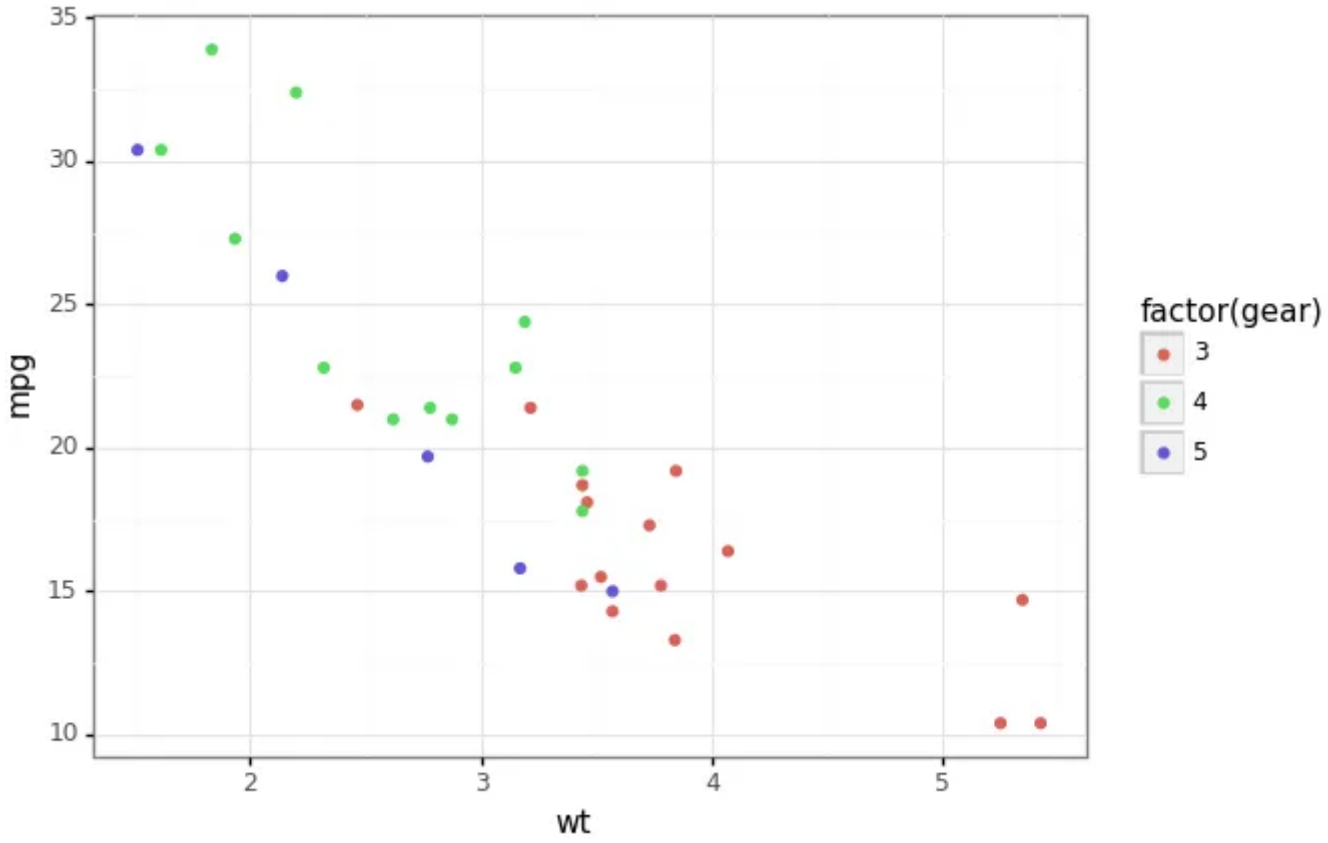
We can clearly see from the above visualization that `mpg` has a negative correlation with the cat `wt` .

Visualize three-dimensions (3-D)

To visualize three dimensions from our dataset, we can leverage color as one of our aesthetic components to visualize one additional dimension besides our other two dimensions as depicted in the following example.

```
(ggplot(mtcars,
            aes('wt', 'mpg', color='factor(gear)'))
            + geom_point()
            + theme_bw())
```

gog_data_viz3.py hosted with ❤ by GitHub view raw



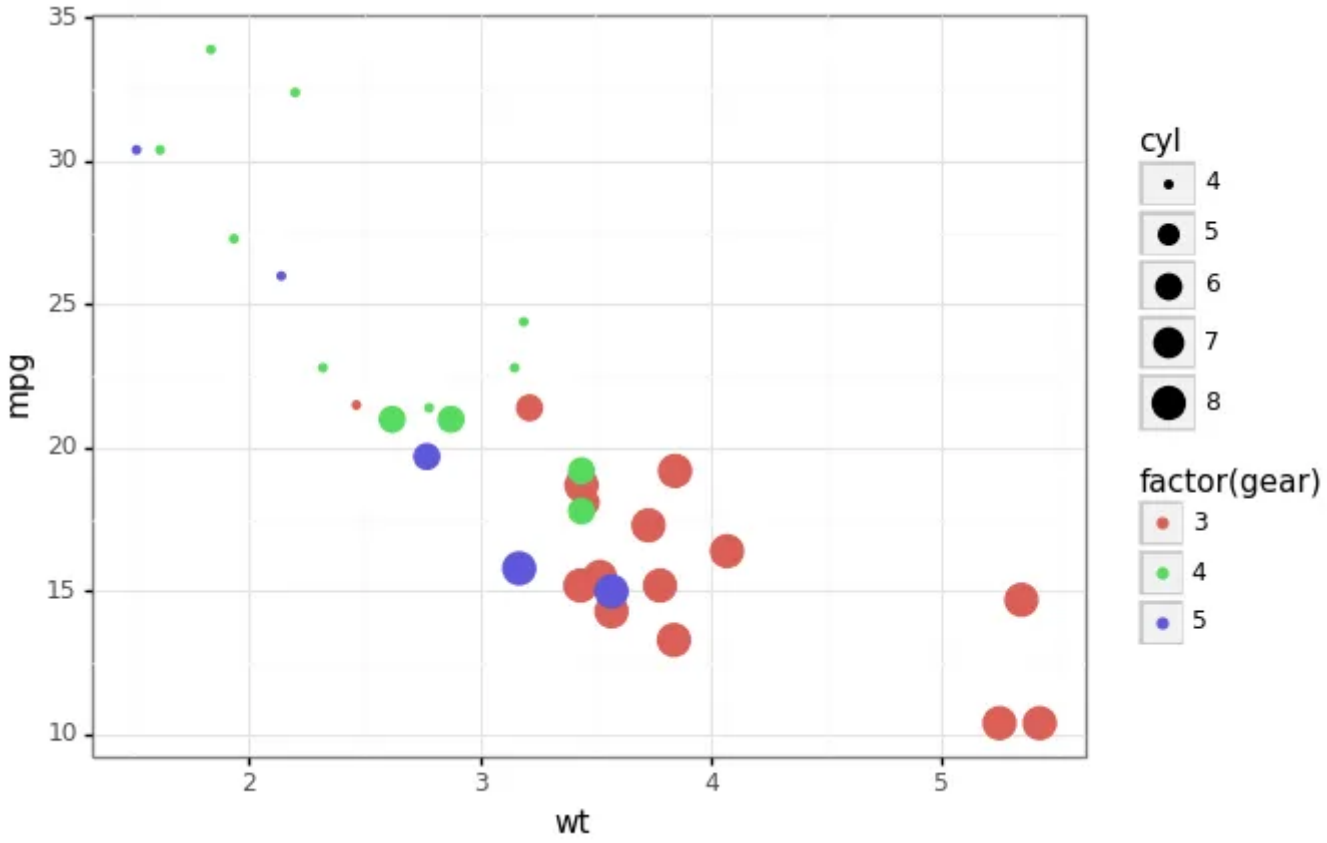
In the above visualization, we depict the cars with different number of gears as separate categories using the color aesthetic along with our other two data dimensions (variables). It is quite clear that cars with a smaller number of **`gears`** on average tend to have higher **`wt`** and lower **`mpg`**.

Visualize four-dimensions (4-D)

To visualize four dimensions from our dataset, we can leverage color as well as size as two of our aesthetics besides other regular components including geoms, data and scale.

```
(ggplot(mtcars,
        aes('wt', 'mpg',
            color='factor(gear)', size='cyl'))
        + geom_point()
        + theme_bw())
```

gog_data_viz4.py hosted with ❤ by GitHub view raw

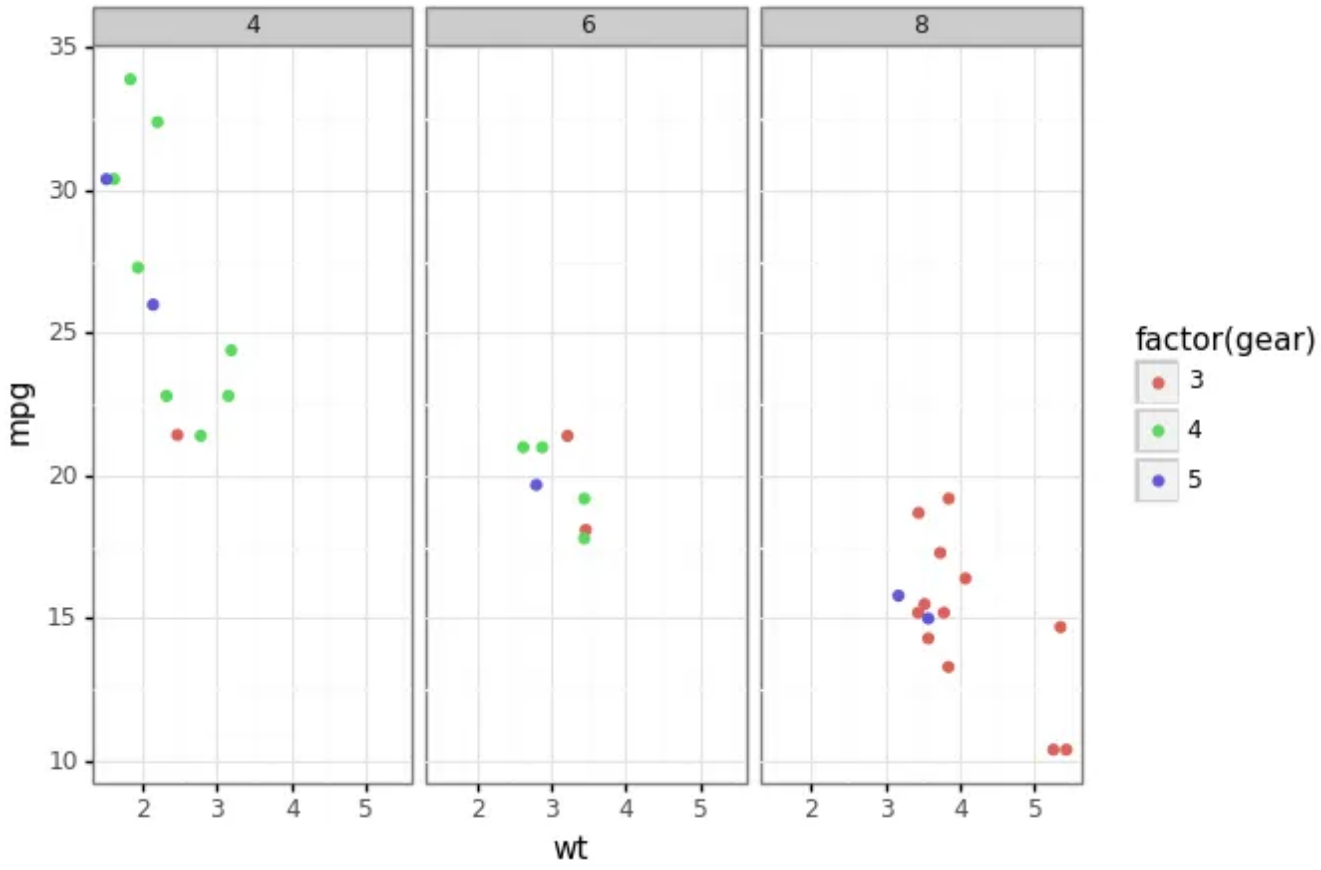


The visualization shows us how powerful aesthetics can be in helping us visualize multiple data dimensions in a single plot. It is quite clear that cars with higher number of **`cyl`** (cylinders) have lower number of **`gears`** and in turn, their **`wt`** is higher and **`mpg`** lower.

Alternatively, we can also use color and facets to depict data in four dimensions instead of size as depicted in the following example.

```
(ggplot(mtcars,
        aes('wt', 'mpg',
            color='factor(gear)'))
        + geom_point()
        + facet_wrap('~cyl')
        + theme_bw())
```

gog_data_viz6.py hosted with ❤ by GitHub view raw



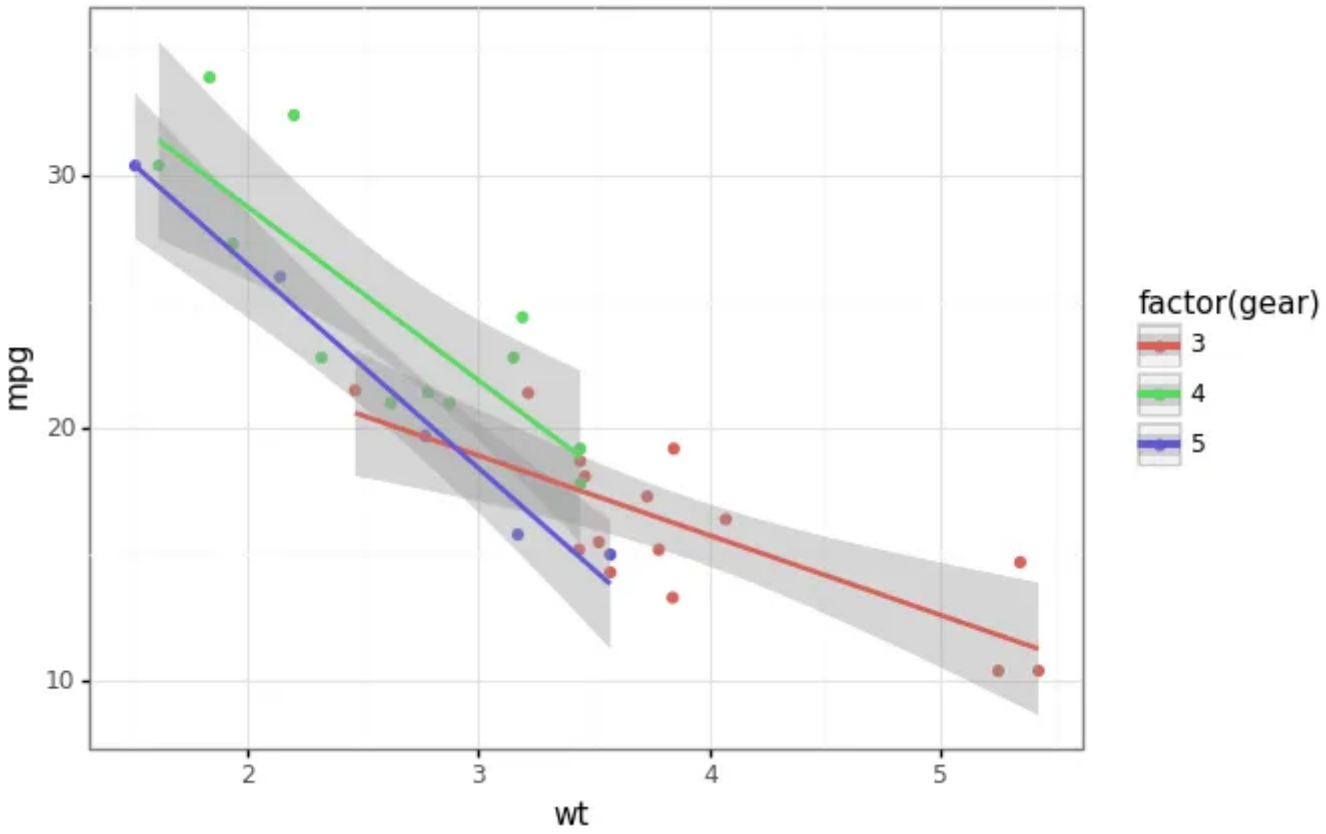
Facets are definitely one of the most powerful components towards building an effective data visualization as shown in the visualization above, where we can clearly see cars with higher `cyl` count have lower `gear` count and similar trends as the previous visualization with color and size.

Visualize data dimensions and statistics

To visualize data dimensions and some relevant statistics (like fitting a linear model), we can leverage statistics along with the other components in our layered grammar.

```
(ggplot(mtcars,
        aes('wt', 'mpg',
            color='factor(gear)'))
        + geom_point()
        + stat_smooth(method='lm')
        + theme_bw())
```

gog_data_viz5.py hosted with ❤ by GitHub view raw



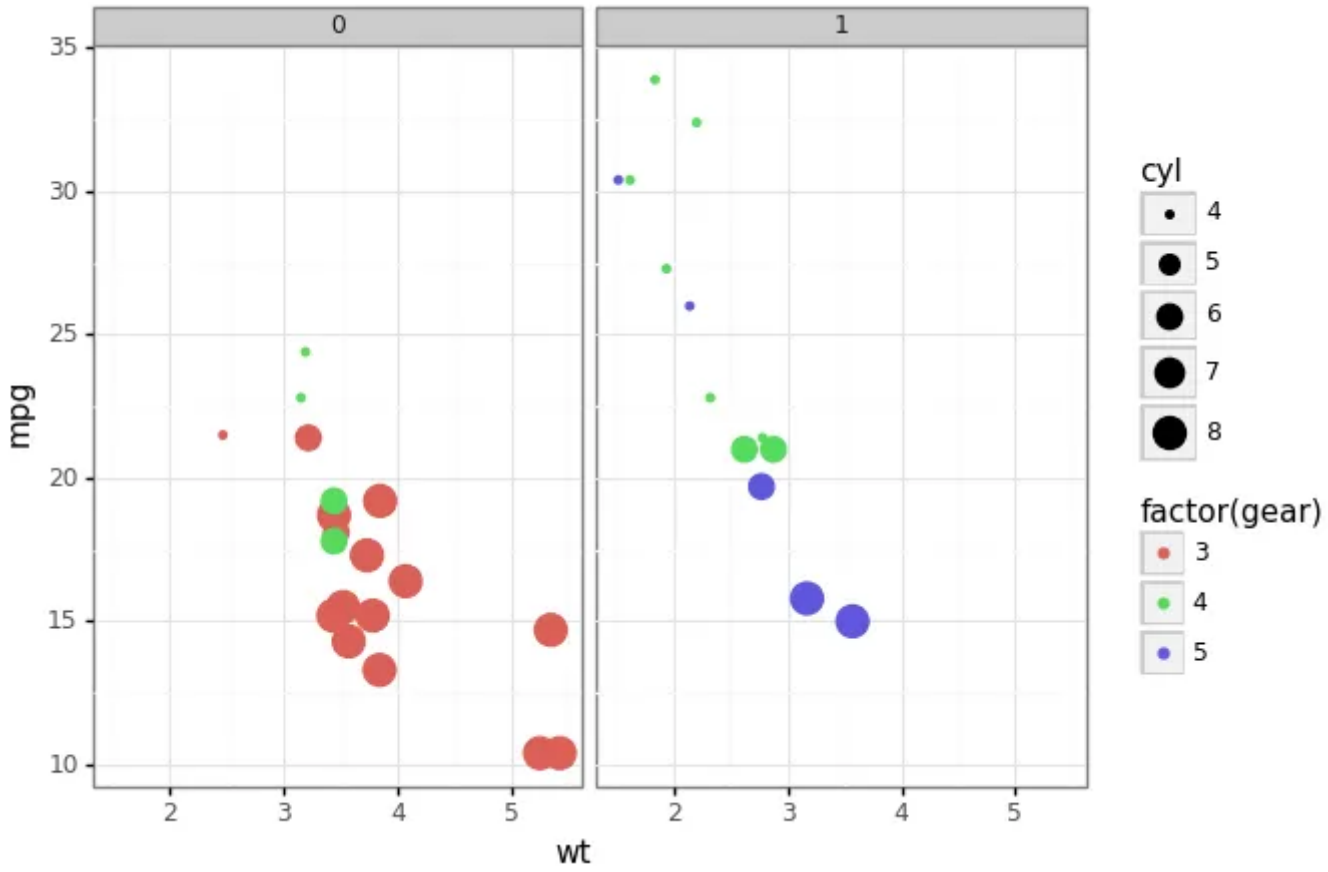
This enables us to see linear model trends for `mpg` based on `wt` due to the statistics component.

Visualize five-dimensions (5-D)

To visualize data in five dimensions, you know the drill by now! We will leverage the power of aesthetics including color, size and facets.

```
(ggplot(mtcars,
        aes('wt', 'mpg',
            color='factor(gear)',
            size='cyl'))
        + geom_point()
        + facet_wrap('~am')
        + theme_bw())
```

gog_data_viz7.py hosted with ❤ by GitHub view raw



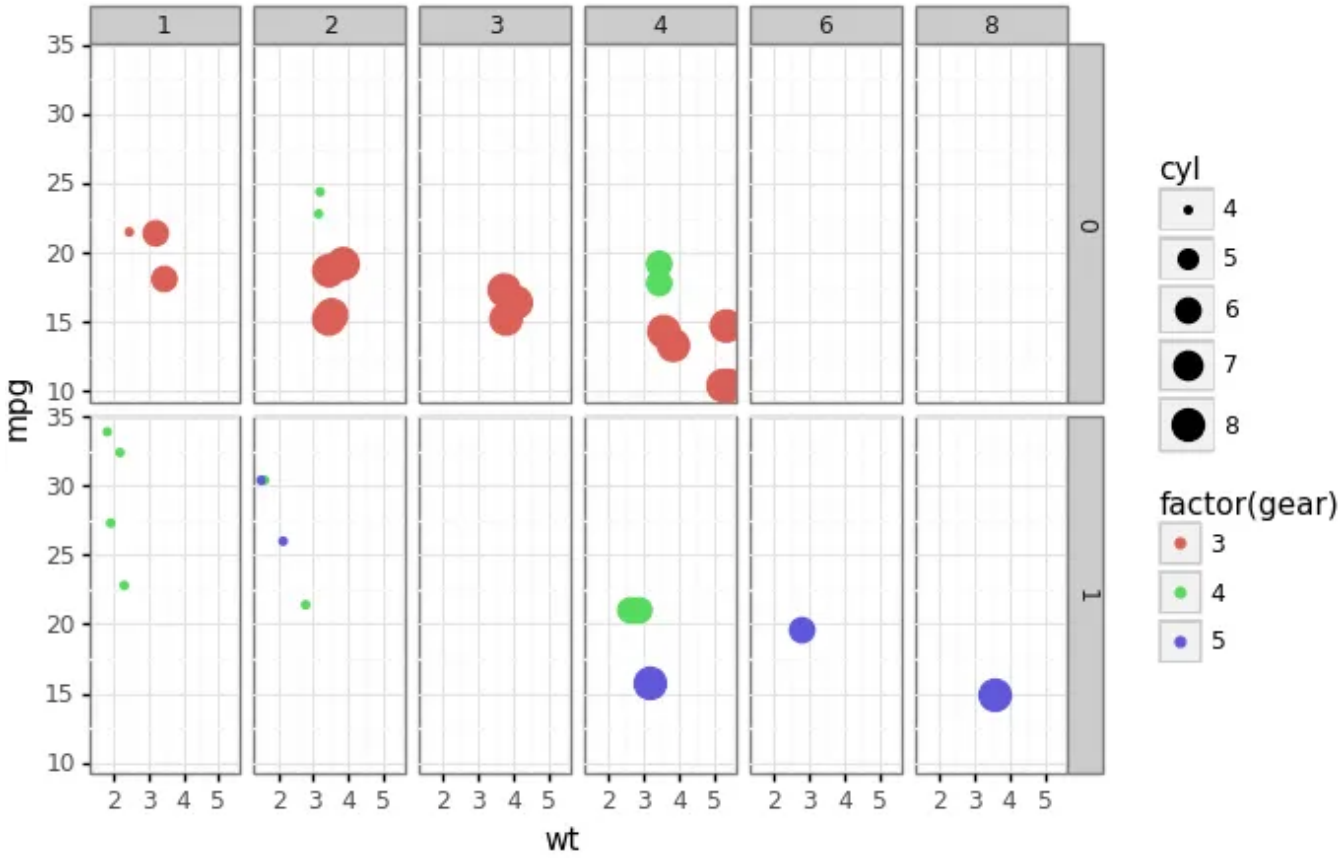
Here we use `am` as the facet where 0 indicates cars with automatic transmission and 1 indicates cars with manual transmission. The plot shows that cars with manual transmission have higher number of gears as compared to cars with automatic transmission. Also, majority of cars with a higher number of cylinders (`cyl`) have automatic transmission. Other insights are similar to what we have observed in the previous plots.

Visualize six-dimensions (6-D)

To visualize data in six dimensions, we can add in an additional facet on the *y-axis* along with a facet on the *x-axis*, and color and size as aesthetics.

```
(ggplot(mtcars,
        aes('wt', 'mpg',
            color='factor(gear)',
            size='cyl'))
        + geom_point()
        + facet_grid('am ~ carb')
        + theme_bw())
```

gog_data_viz8.py hosted with ❤ by GitHub view raw

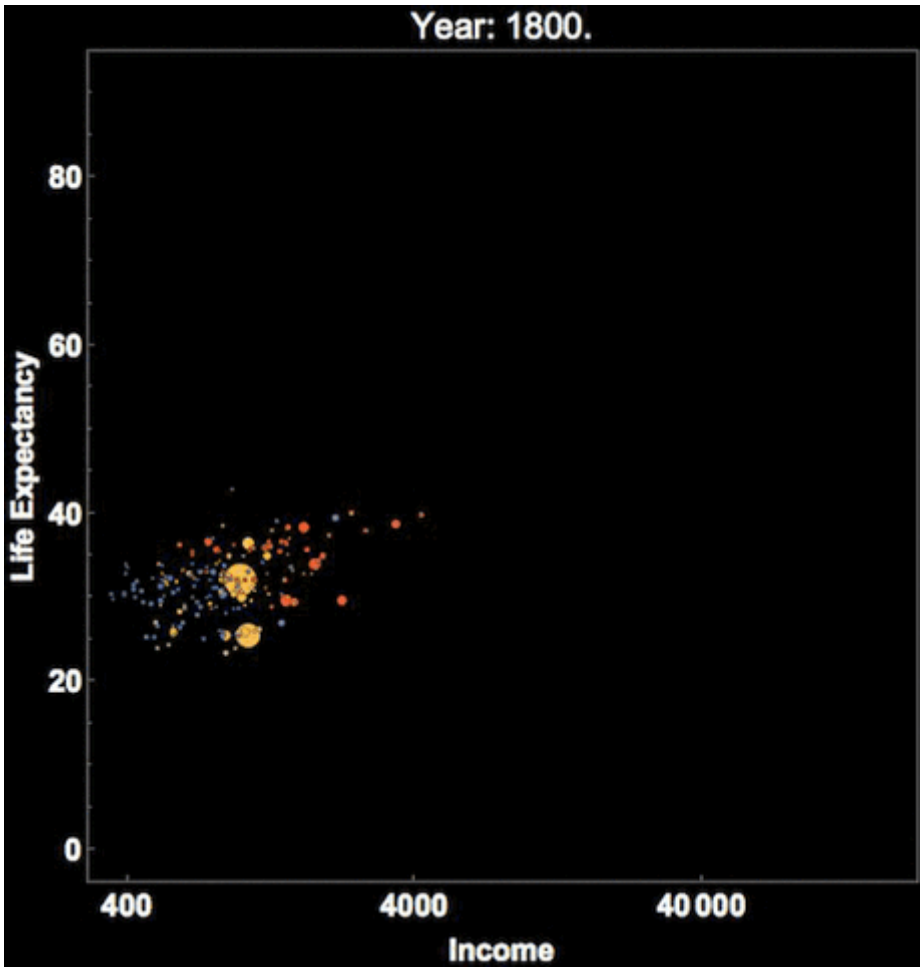


We represent transmission (`am`) as 0 (automatic) and 1 (manual) as a facet on the *y-axis* and number of carburetors (`carb`) as a facet on the *x-axis* besides the other dimensions represented using other aesthetics similar to our previous plots. An interesting insight from the above visualization is that cars with higher number of gears have manual transmission (`am`) and higher number of carburetors (`carb`). Do you notice any other interesting insights?

Can we go higher?

The pressing question is, can we go higher than six dimensions? Well, it definitely becomes more and more difficult to hack our way around the limitations of a two-dimensional rendering device to visualize more data dimensions.

One method is to use more facets and subplots. Besides this, you can also use the notion of time if your dataset has a temporal aspect as depicted in the following example.



Hans Rosling's famous visualization of global population, health and economic indicators

This depicts ***Hans Rosling's*** famous visualization of depicting global population, health and various economic indicators across all countries. This was also presented in an **official TED conference** which I would recommend everyone to check out if they haven't done it already!

This should give you a good perspective on how to leverage the layered grammar of graphics to visualize multi-dimensional data.

Conclusion

Like I have mentioned time and again, data visualization is an art as well as a science. This article should give you enough motivation and examples to get started with understanding and leveraging a layered grammar of graphics framework towards building effective visualizations of your own on multi-dimensional data.

All the code used in this article is available as a **Jupyter notebook** along with other content and slides in **my GitHub repository**.

dipanjanS/art_of_data_visualization

The art of effective visualization of multi-dimensional data - dipanjanS/art_of_data_visualization

github.com

I do cover how to visualize multi-dimensional data using state-of-the-art data visualization frameworks in Python like `seaborn` and `matplotlib` on slightly more complex data. The ***following article*** should help you get started on the same if you are interested.

The Art of Effective Visualization of Multi-dimensional Data

Strategies for Effective Data Visualization

towardsdatascience.com

A major portion of these articles was covered in one of my recent conference talks in **ODSC, 2018**. You can check out the **full talk agenda and slides here**. I will post the YouTube conference talk once it's out!

The Art of Effective Visualization of Multi-dimensional Data - A hands-on Approach - ODSC India...

Descriptive Analytics is one of the core components of any analysis life-cycle pertaining to a data science project or...

confengine.com

Have feedback for me? Or interested in working with me on research, data science, artificial intelligence or even publishing an article on TDS? You can reach out to me on **LinkedIn**.

Dipanjan Sarkar - Data Scientist - Intel Corporation | LinkedIn

View Dipanjan Sarkar's profile on LinkedIn, the world's largest professional community. Dipanjan has 6 jobs listed on...

www.linkedin.com

Thanks to *Durba* for editing this article.

Data Visualization | Python | Data Science | Machine Learning

Following

Written by Dipanjan (DJ) Sarkar

10.5K Followers · Writer for Towards Data Science

Data Science Lead

More from Dipanjan (DJ) Sarkar and Towards Data Science