COMP4010 - Week 1

2024-02-20

Week 1

Application Exercises

Task 1.

- Data:
- Mapping:
- Statistical transformation:
- Geometric object:
- Position adjustment:
- Scale:
- Coordinate system:
- Faceting:

Task 2.

```
# Your code here
```

Task 3.

```
# Your code here
```

Reading Material

Hello World! but in R

Create a chunk below and create a vector of numbers and calculate its mean. (This is akin to printing 'Hello World' in other languages, statisticians are not as fun :D)

```
myVector <- c(1,2,3,4)
mean(myVector)
```

```
## [1] 2.5
```

Using CRAN to install ggplot2

Alternatively, you can just run this in the console below.

```
#install.packages("ggplot2") # uncomment to install
library(ggplot2) # Import
```

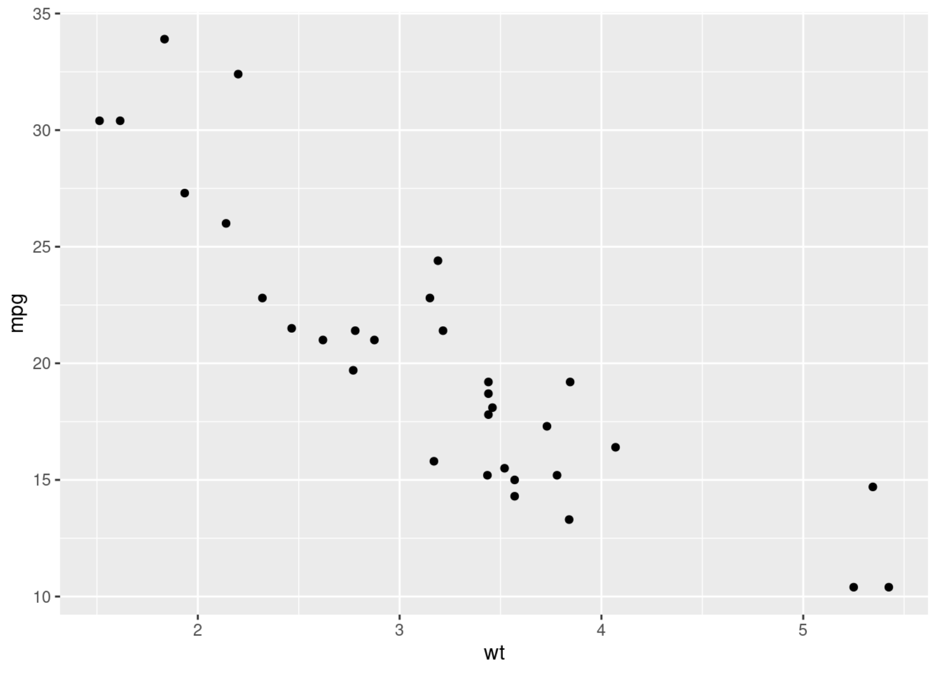
Hello to ggplot2

This uses the built-in `mtcars` dataset. To preview this dataset, you can use `summary(mtcars)` or `View(mtcars)`.

```
View(mtcars)
```

We can see that there are the `mpg` (miles per gallons) and `wt` (weight) columns in the dataset. Let's plot the 2 dimensions on the scatterplot using `ggplot2`.

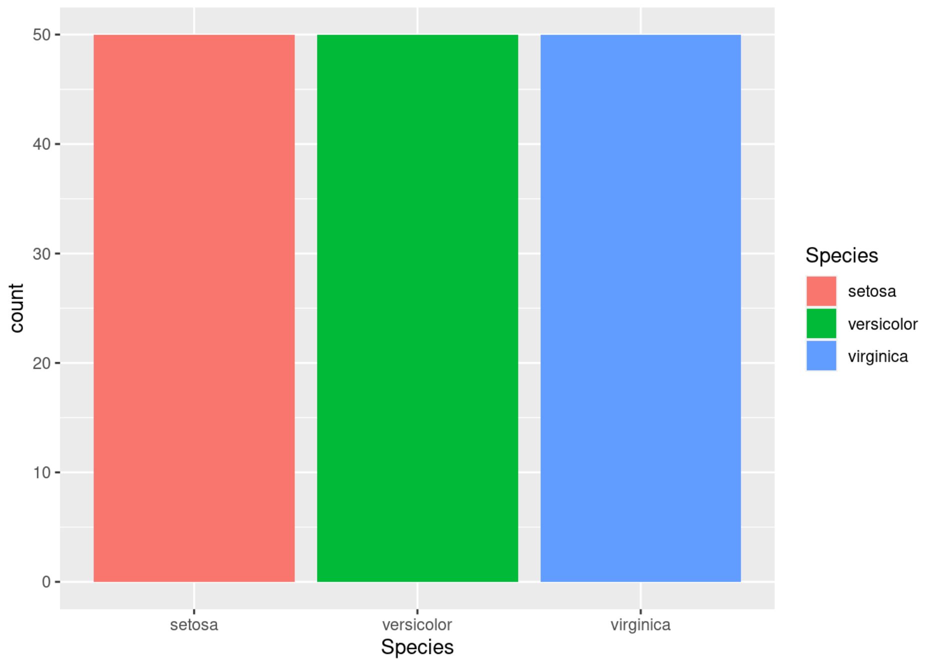
```
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point()
```



Bar chart with iris dataset

We can also try making a bar chart with the built-in `iris` dataset.

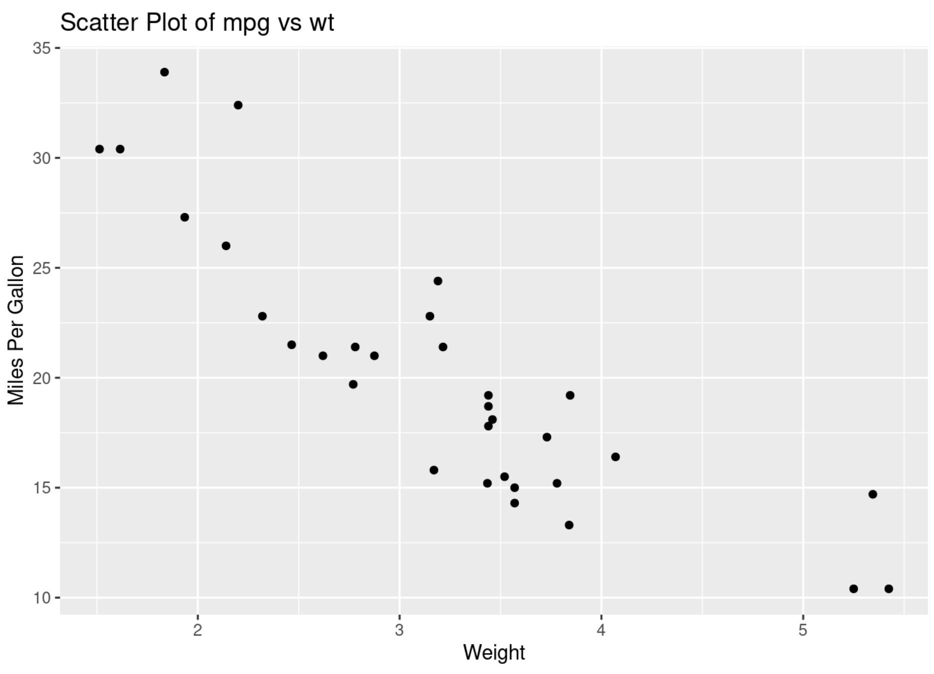
```
ggplot(data = iris, aes(x = Species, fill = Species)) + geom_bar()
```



Customizing plots

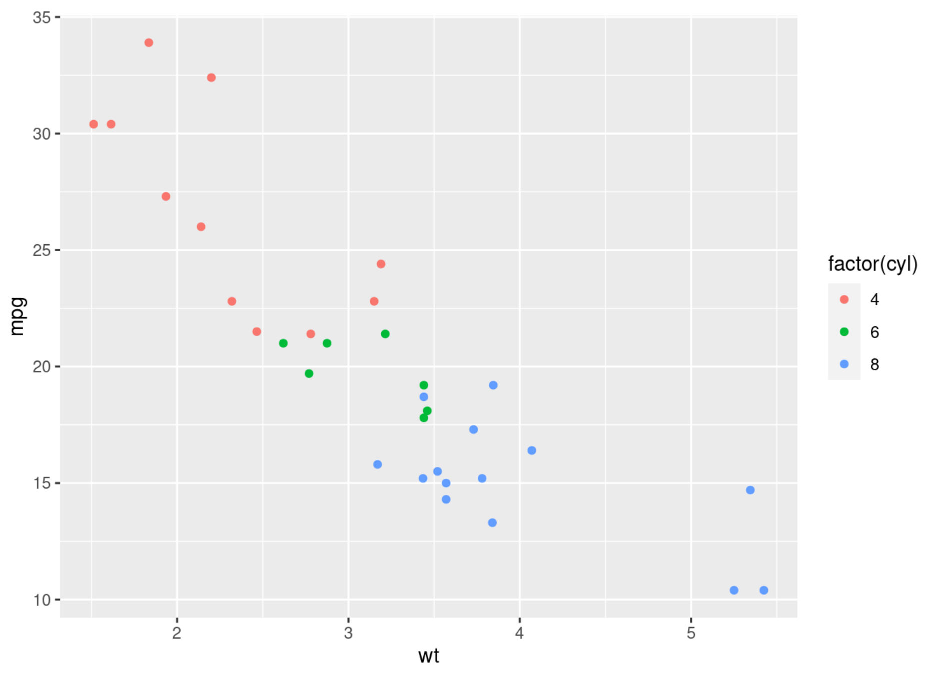
Adding titles and labels:

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  ggtitle("Scatter Plot of mpg vs wt") +
  xlab("Weight") +
  ylab("Miles Per Gallon")
```



Changing colors:

```
ggplot(data = mtcars, aes(x = wt, y = mpg, color = factor(cyl))) +
  geom_point()
```

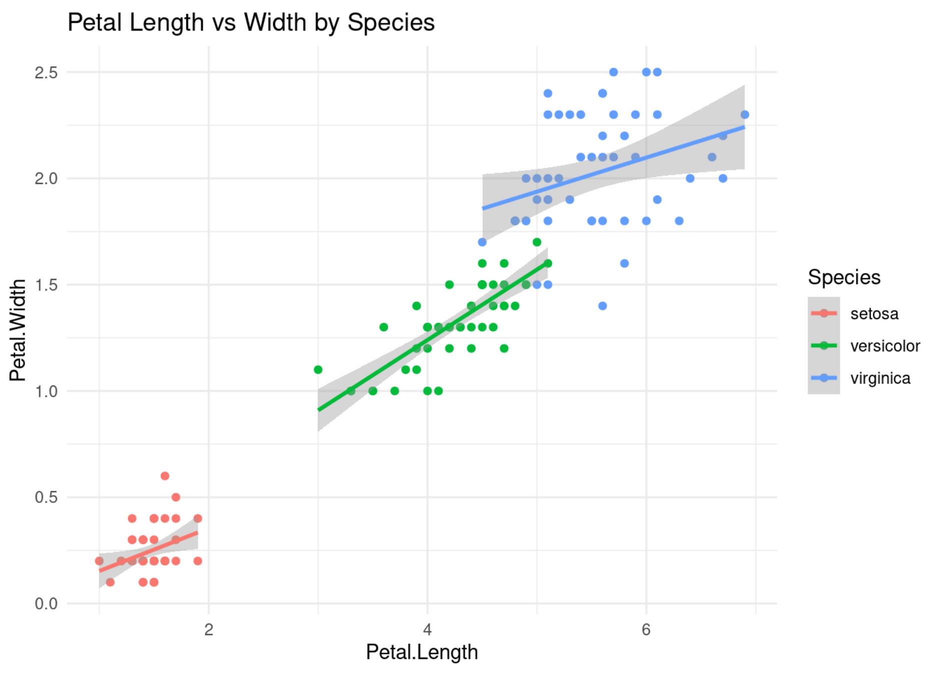


Experimenting with the Iris Dataset

- **Dataset**: Iris (available in R by default)
- **Task**: Create a scatter plot showing the relationship between petal length and petal width, colored by species.
- **Customization**: Add a smooth regression line for each species.

```
ggplot(iris, aes(x = Petal.Length, y = Petal.Width, color = Species)) +
    geom_point() +
    geom_smooth(method = "lm") +
    ggtitle("Petal Length vs Width by Species") +
    theme_minimal()
```

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



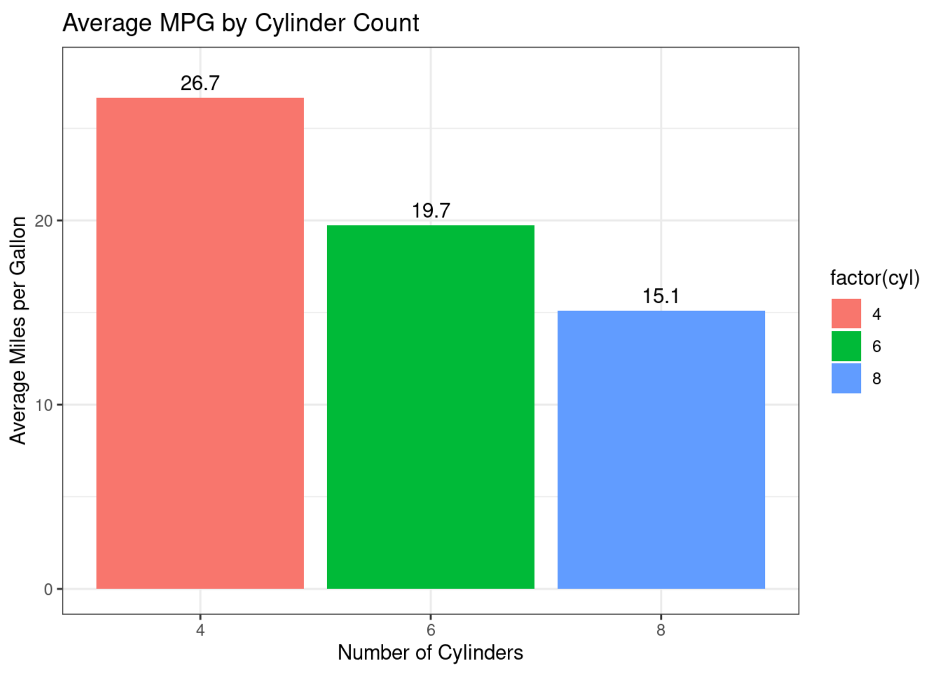
Visualizing the mtcars Dataset

- **Dataset**: mtcars (available in R by default)
- **Task**: Create a bar plot showing the average miles per gallon (mpg) for cars with different numbers of cylinders.
- **Customization**: Use a different fill color for each cylinder type and add labels for the average mpg.

```
ggplot(mtcars, aes(x = factor(cyl), y = mpg, fill = factor(cyl))) +
    geom_bar(stat = "summary", fun = mean) +
    geom_text(stat = 'summary', aes(label = round(..y.., 1)), vjust = -0.5) +
    labs(x = "Number of Cylinders", y = "Average Miles per Gallon", title = "Average MPG by Cylinder Count") +
    theme_bw()
```

```
## Warning: The dot-dot notation (`..y..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(y)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

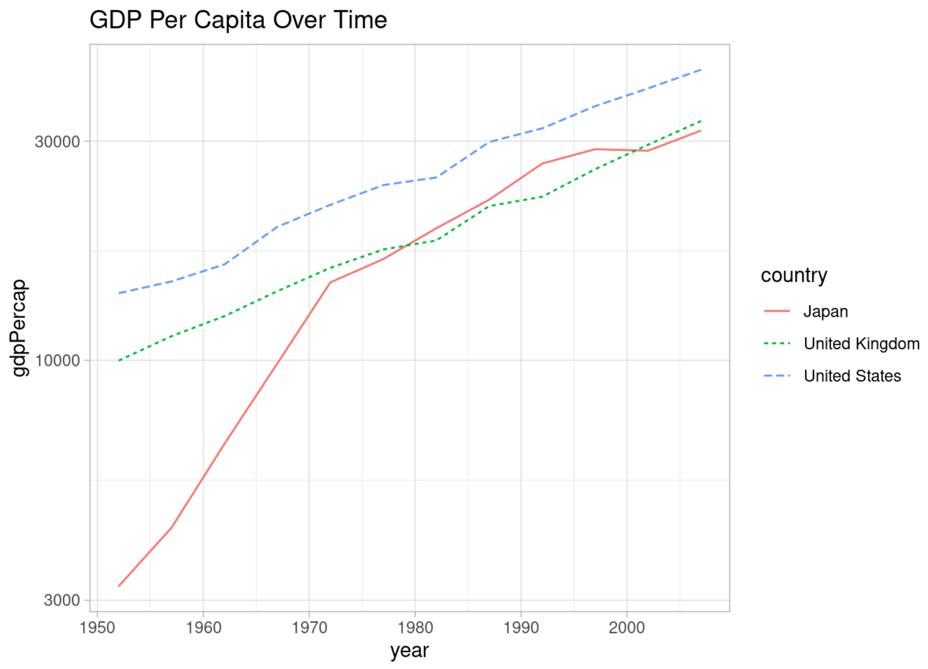
```
## No summary function supplied, defaulting to `mean_se()`
```



Exploring the gapminder Dataset

- **Dataset**: gapminder (install using `install.packages("gapminder")` and then `library(gapminder)`)
- **Task**: Create a line plot showing GDP per capita over time for select countries.
- **Customization**: Use different line types and colors for each country.

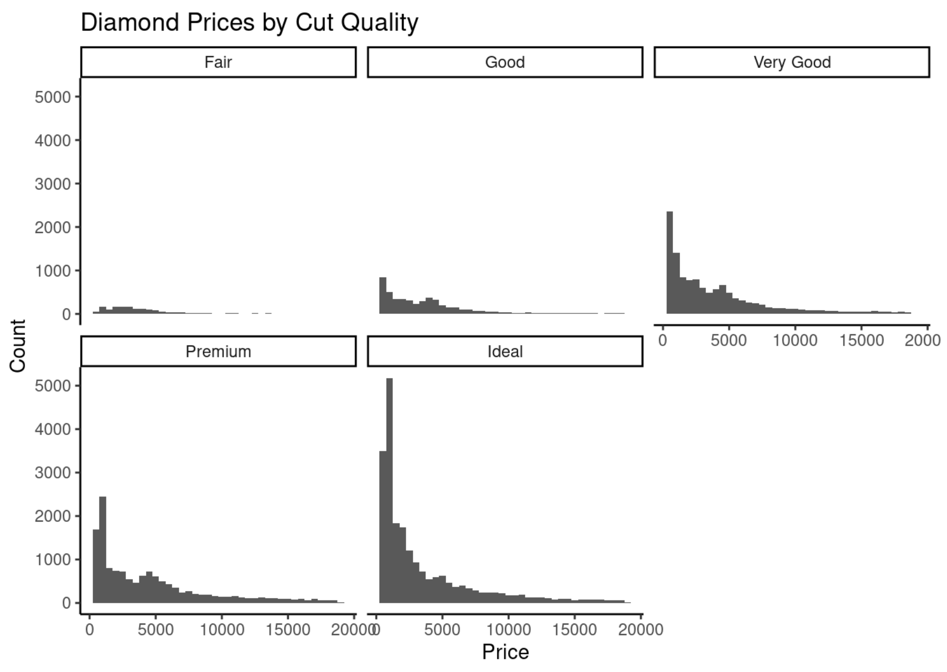
```
# install.packages("gapminder")
library(gapminder)
ggplot(subset(gapminder, country %in% c("Japan", "United Kingdom", "United States")),
       aes(x = year, y = gdpPercap, color = country, linetype = country)) +
    geom_line() +
    scale_y_log10() +
    ggtitle("GDP Per Capita Over Time") +
    theme_light()
```



Working with the diamonds Dataset

- **Dataset**: diamonds (part of ggplot2 package)
- **Task**: Create a histogram of diamond prices, faceted by cut quality.
- **Customization**: Adjust the bin width and use a theme that enhances readability.

```
ggplot(diamonds, aes(x = price)) +
    geom_histogram(binwidth = 500) +
    facet_wrap(~cut) +
    labs(title = "Diamond Prices by Cut Quality", x = "Price", y = "Count") +
    theme_classic()
```


Diamond Prices by Cut Quality
Fair
Good
Very Good
Premium
Ideal
Count
Price