



# Week 10: *Data Visualization*

🏛️ EMSE 4571 / 6571: Intro to Programming for Analytics

👤 John Paul Helveston

📅 April 03, 2025

# Quiz 8

10:00

Write your name on the quiz!

## Rules:

- Work alone; no outside help of any kind is allowed.
- No calculators, no notes, no books, no computers, no phones.

# Week 10: *Data Visualization*

1. Plotting with Base R

2. Plotting with ggplot2: Part 1

**BREAK**

3. Plotting with ggplot2: Part 2

4. Tweaking your ggplot

# Week 10: *Data Visualization*

1. Plotting with Base R

2. Plotting with ggplot2: Part 1

BREAK

3. Plotting with ggplot2: Part 2

4. Tweaking your ggplot

# Today's data:

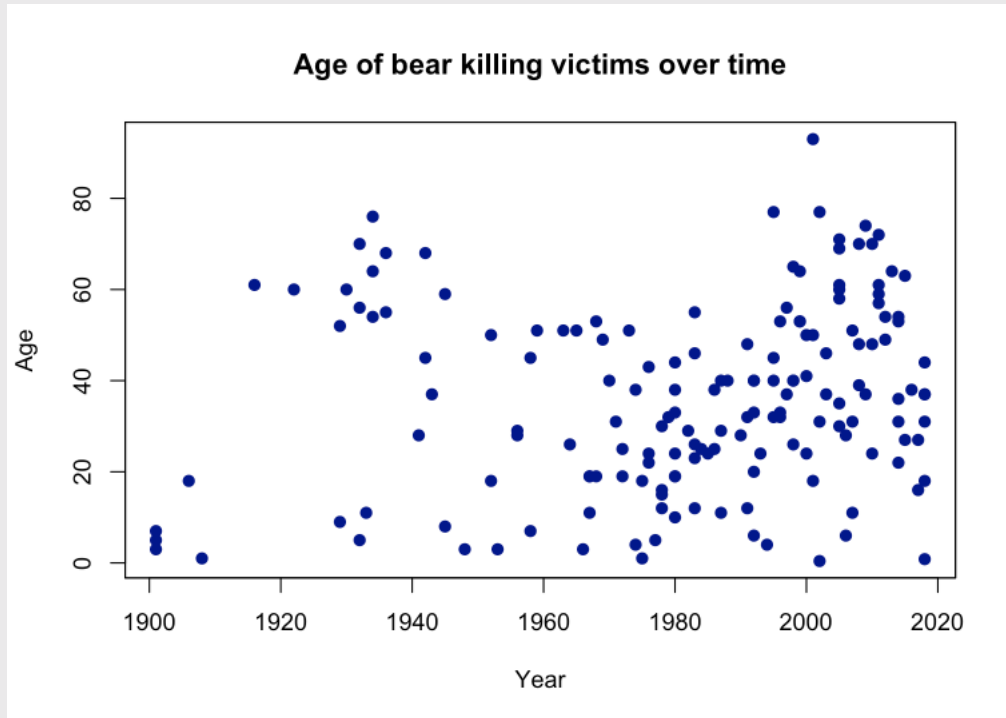
## Bear attacks in North America

Explore the `bears` data frame:

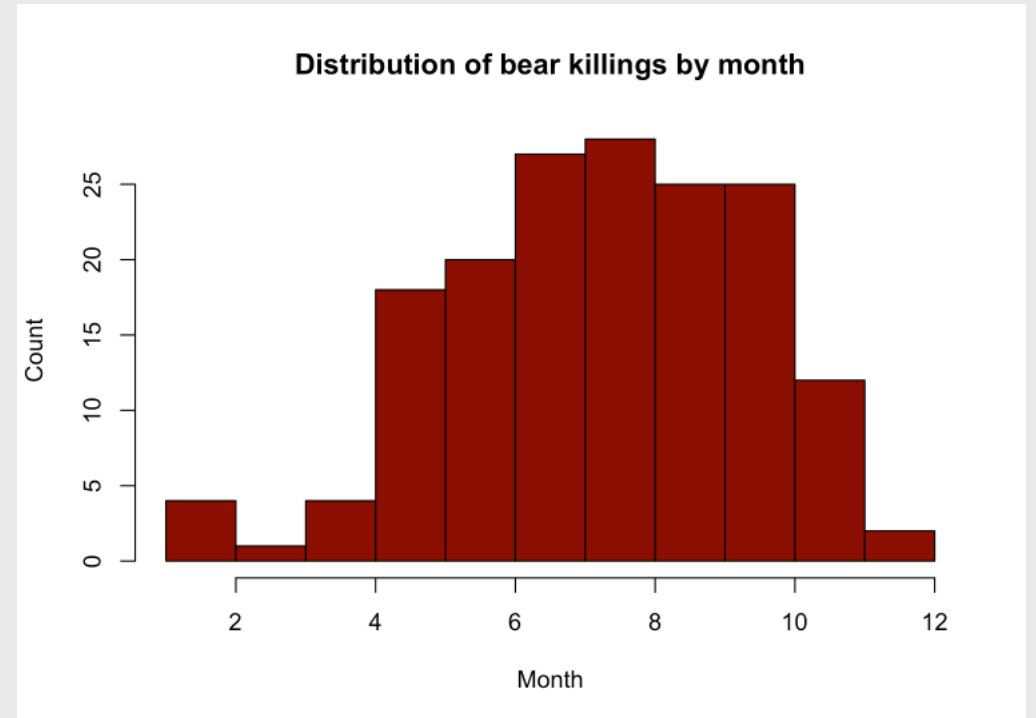
```
glimpse(bears)  
head(bears)
```

# Two basic plots in R

## Scatterplots



## Histograms



# Scatterplots with `plot()`

Plot relationship between two variables

General syntax:

```
plot(x = x_vector, y = y_vector)
```

# Scatterplots with `plot()`

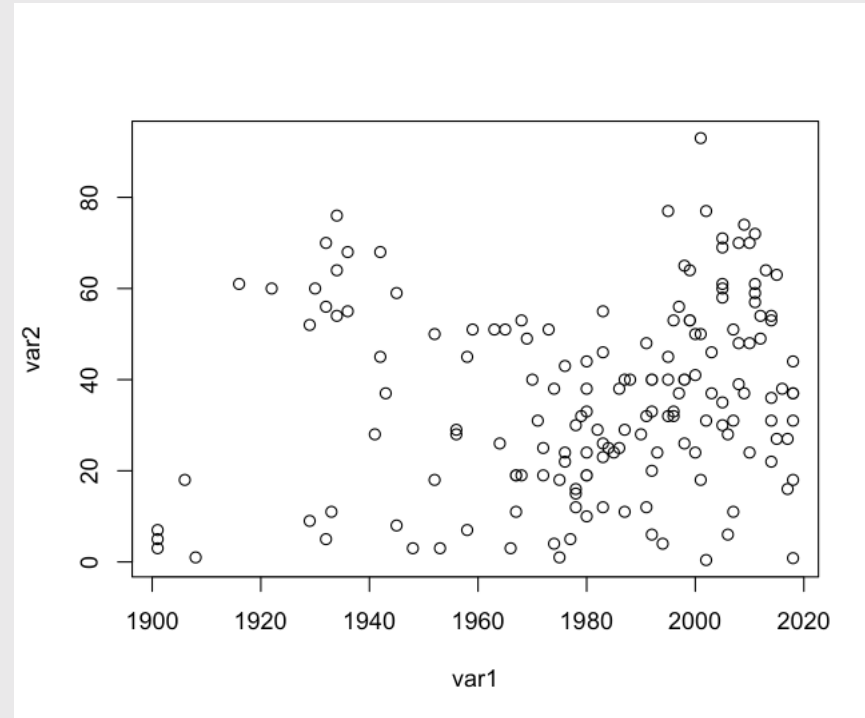
Plot relationship between two variables

General syntax:

```
plot(x = x_vector, y = y_vector)
```

Example:

```
var1 <- bears$year  
var2 <- bears$age  
plot(x = var1, y = var2)
```





# Scatterplots with `plot()`

`x` and `y` must have the same length!

```
var2 <- var2[-1]
```

```
length(var1) == length(var2)
```

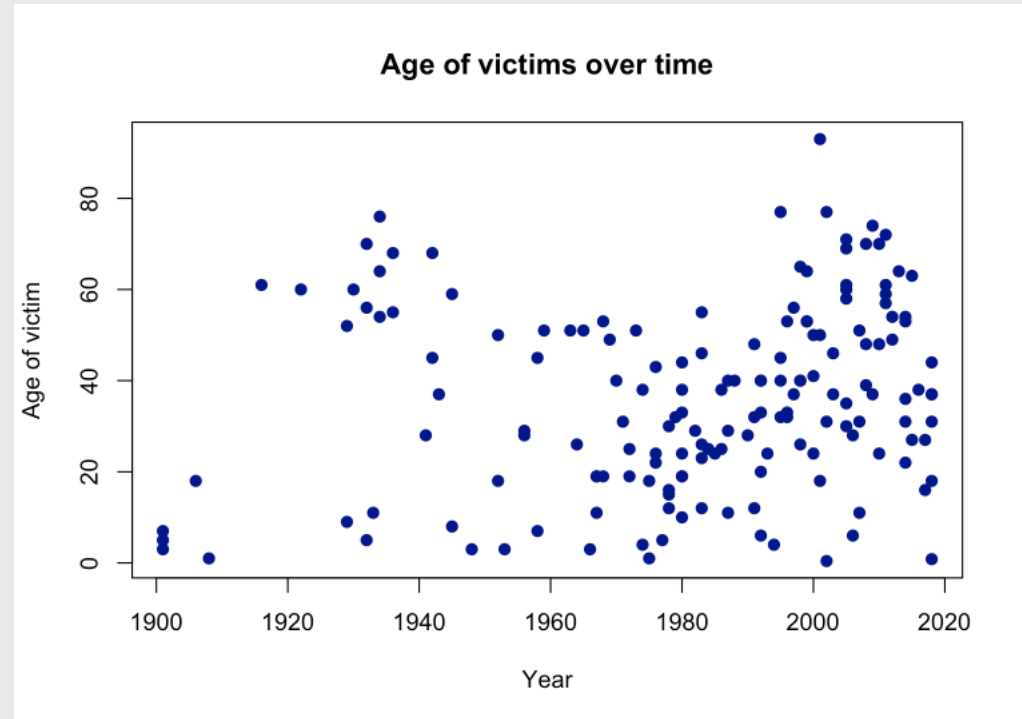
```
#> [1] FALSE
```

```
plot(x = var1, y = var2)
```

```
#> Error in xy.coords(x, y, xlabel, ylabel, log): 'x' and 'y' lengths differ
```

# Making `plot()` pretty

```
plot(  
  x = bears$year,  
  y = bears$age,  
  col = 'darkblue', # Point color  
  pch = 19, # Point shape  
  main = "Age of victims over time",  
  xlab = "Year",  
  ylab = "Age of victim"  
)
```



10:00

Your turn: `plot()`

Does the annual number of bird impacts appear to be changing over time?

Make a plot using the `birds` data frame to justify your answer.

Hint: You may need to create a *summary* data frame to answer this question!

**Bonus:** Make your plot pretty!

# Histograms with `hist()`

Plot the *distribution* of a single variable

General syntax:

```
hist(x = x_vector)
```

# Histograms with `hist()`

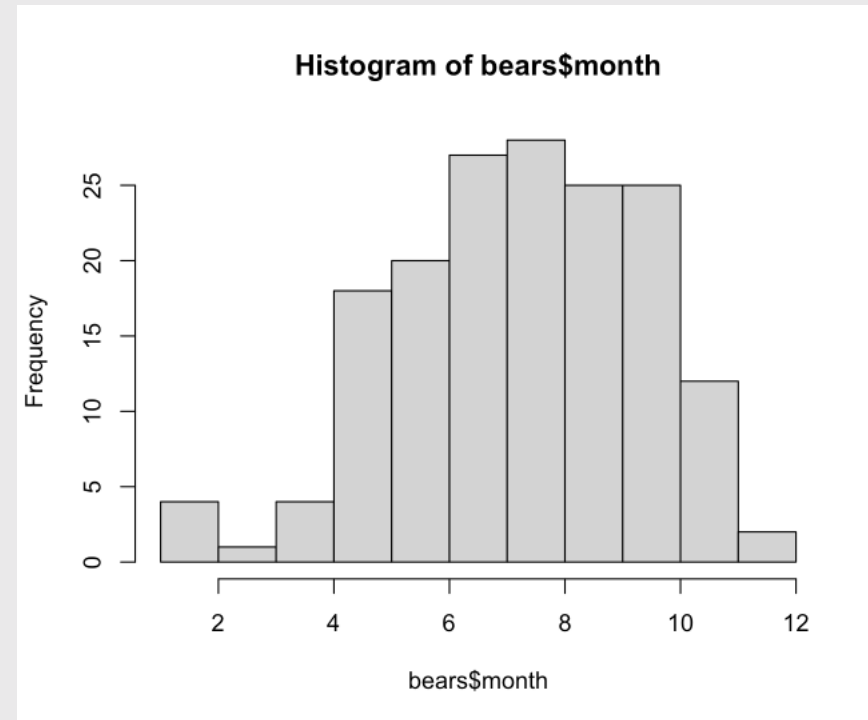
Plot the *distribution* of a single variable

General syntax:

```
hist(x = x_vector)
```

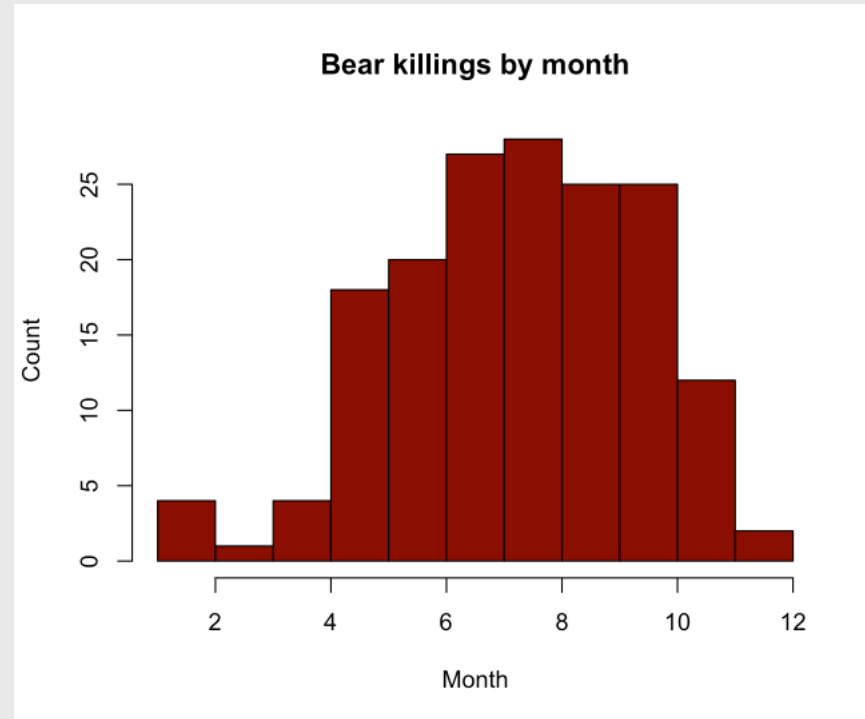
Example:

```
hist(bears$month)
```



# Making `hist()` pretty

```
hist(  
  x      = bears$month,  
  breaks = 12,  
  col    = 'darkred',  
  main   = "Bear killings by month",  
  xlab   = "Month",  
  ylab   = "Count"  
)
```



10:00

## Your turn: `hist()`

Make plots using the `birds` data frame to answer these questions

1. Which months have the highest and lowest number of bird impacts in the dataset?
2. Which aircrafts experience more impacts: 2-engine, 3-engine, or 4-engine?
3. At what height do most impacts occur?

**Bonus:** Make your plots pretty!

# Week 10: *Data Visualization*

1. Plotting with Base R

2. Plotting with ggplot2: Part 1

BREAK

3. Plotting with ggplot2: Part 2

4. Tweaking your ggplot



# Better figures with `ggplot2`



Art by [Allison Horst](#)

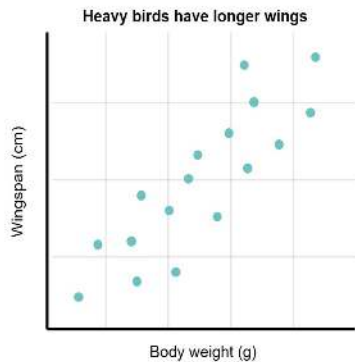
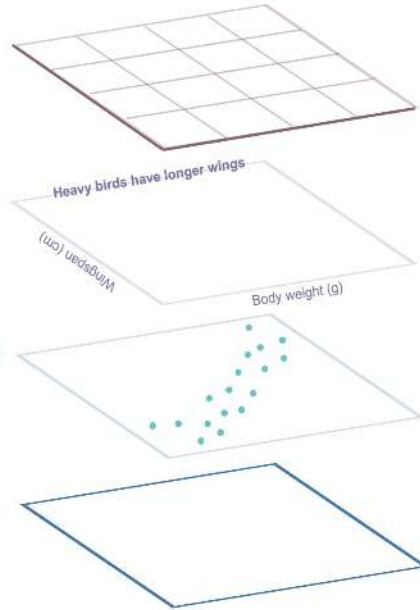
## MAKING A GRAPH WITH GGPLOT2

Customise the look of your plot with themes  
(pre-made or your own!):  
`+ theme_bw()`

Add labels and titles:  
`+ labs(x = "Body weight (g)", y = "Wingspan (cm)",  
title = "Heavy birds have longer wings")`

Specify the type of graph and the variables to use:  
`+ geom_point(aes(x = body.weight, y = wingspan))`

Plot the device containing your data:  
`ggplot(data = birds)`



# "Grammar of Graphics"

Concept developed by Leland Wilkinson  
(1999)

**ggplot2** package developed by Hadley  
Wickham (2005)

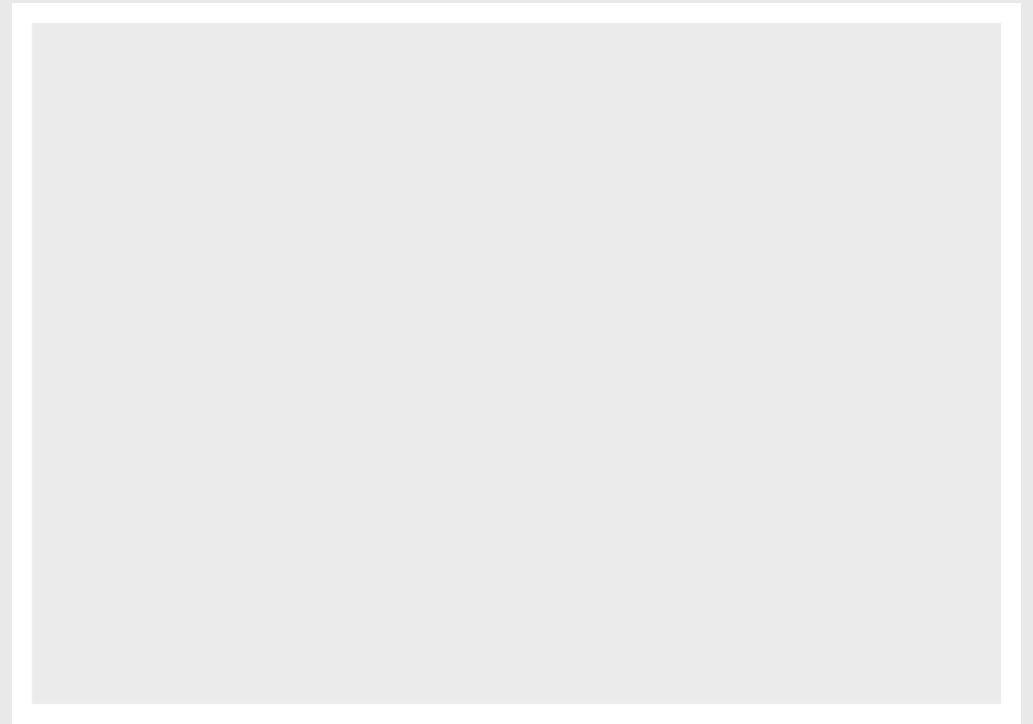
# Making plot layers with ggplot2

1. The data (we'll use `bears`)
2. The aesthetic mapping (what goes on the axes?)
3. The geometries (points? bars? etc.)

# Layer 1: The data

The `ggplot()` function initializes the plot with whatever data you're using

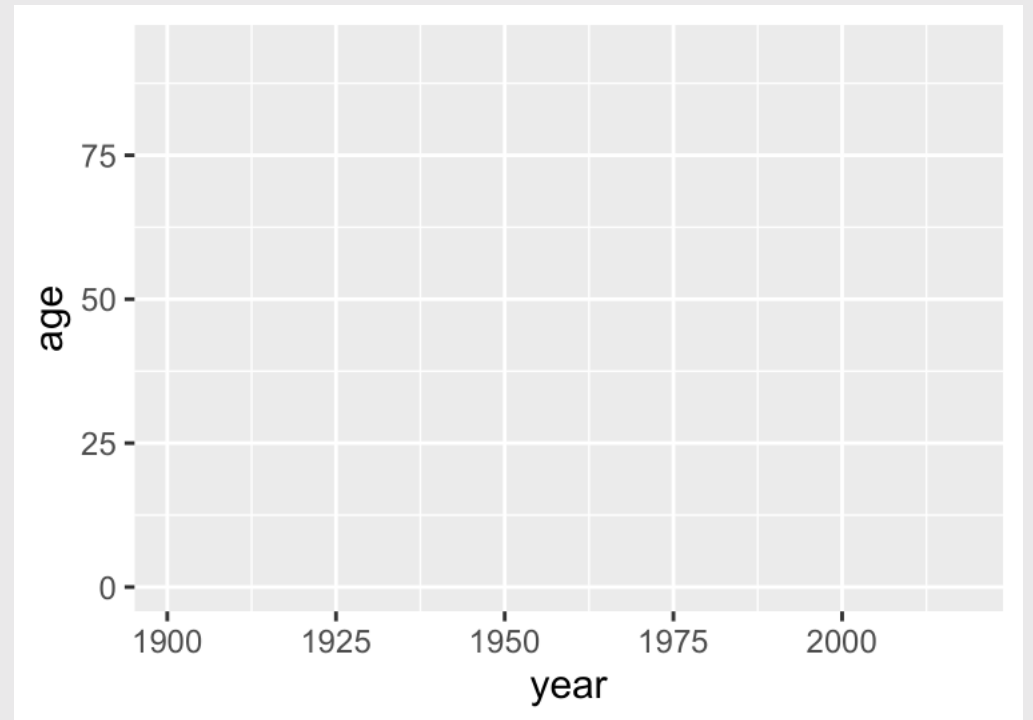
```
ggplot(data = bears)
```



# Layer 2: The aesthetic mapping

The `aes()` function determines which variables will be *mapped* to the geometries (e.g. the axes)

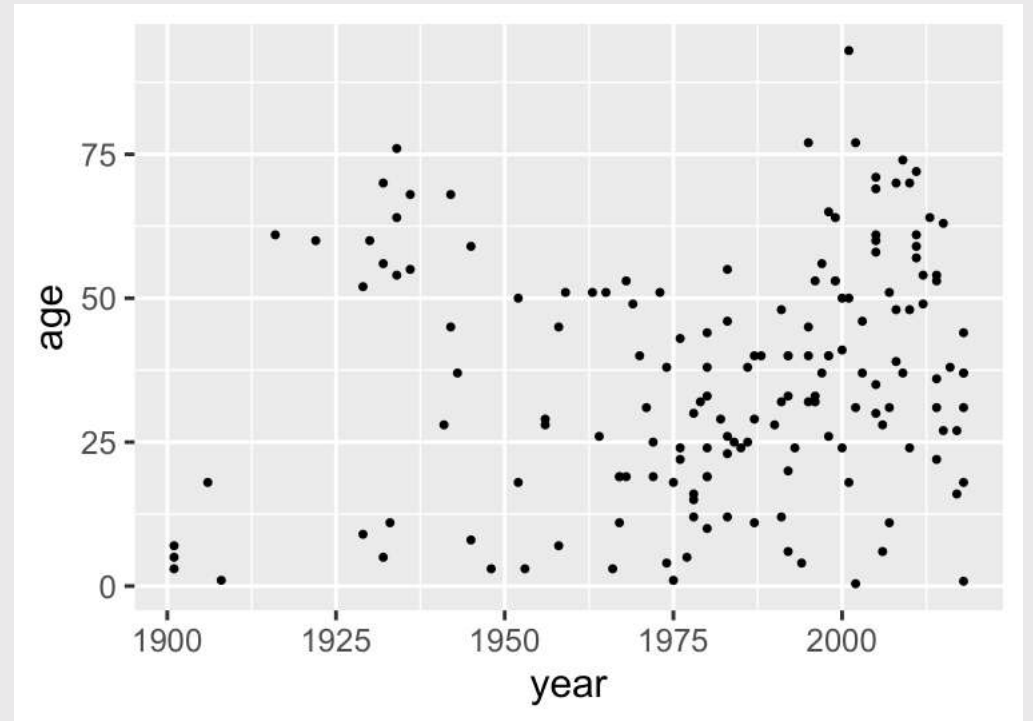
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age))
```



# Layer 3: The geometries

Use `+` to add geometries (e.g. points)

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point()
```



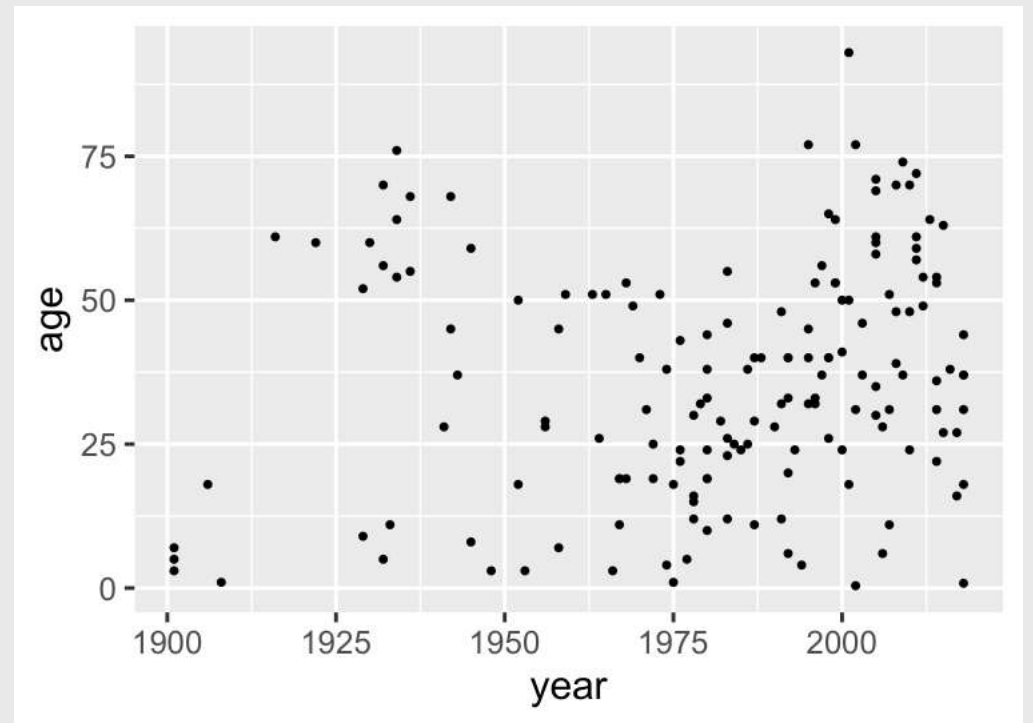
# Other common geometries

- `geom_point()`: scatter plots
- `geom_line()`: lines connecting data points
- `geom_col()`: bar charts
- `geom_boxplot()`: boxes for boxplots

# Scatterplots with `geom_point()`

Add points:

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point()
```

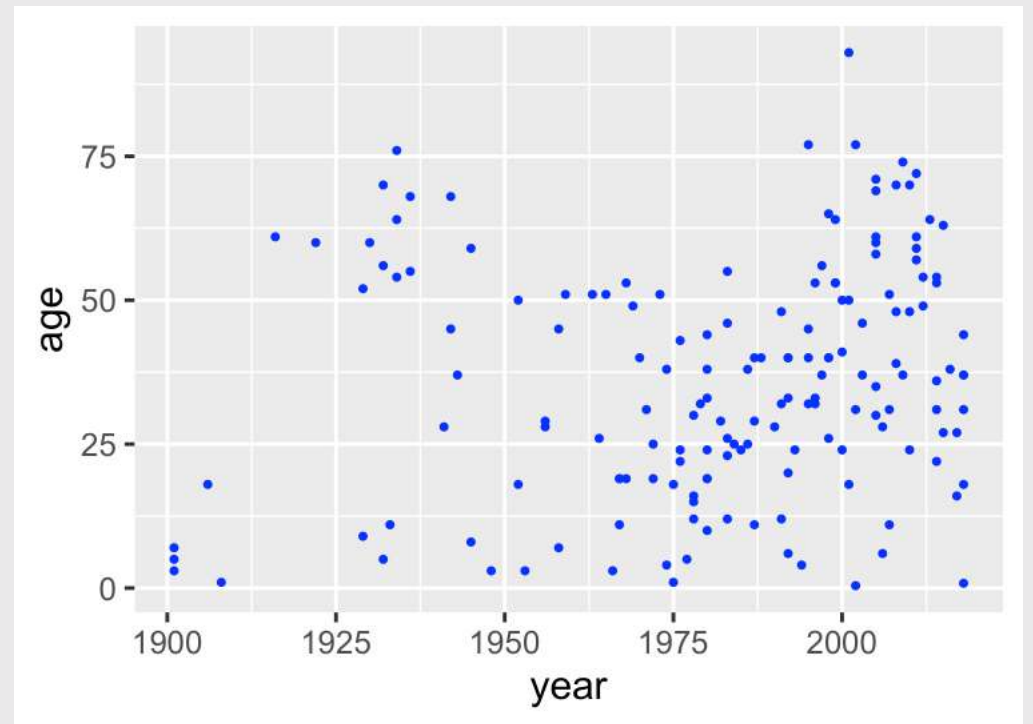




# Scatterplots with `geom_point()`

Change the color of all points:

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point(color = 'blue')
```

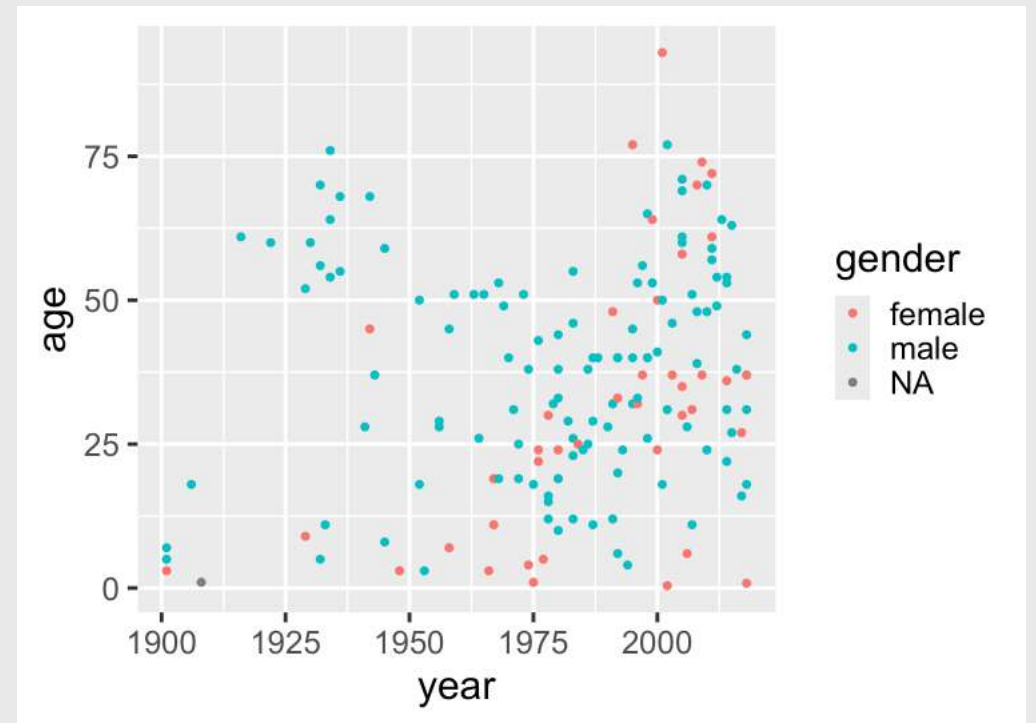


# Scatterplots with `geom_point()`

Map the point color to a **variable**:

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point(aes(color = gender))
```

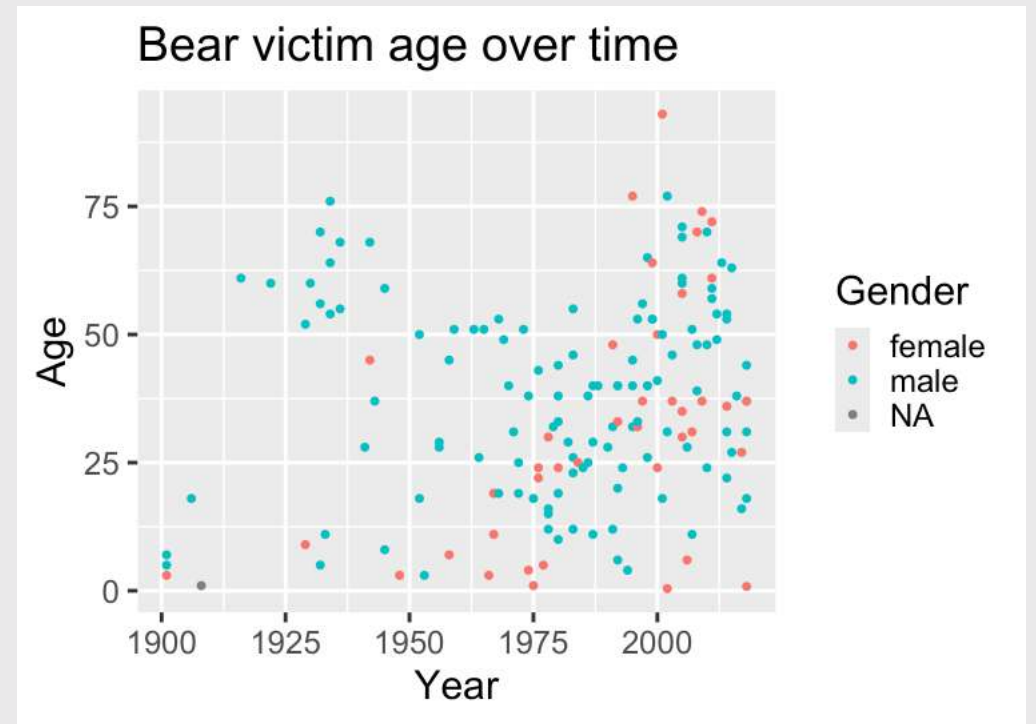
Note that `color = gender` is *inside* `aes()`



# Scatterplots with `geom_point()`

Adjust labels with `labs()` layer:

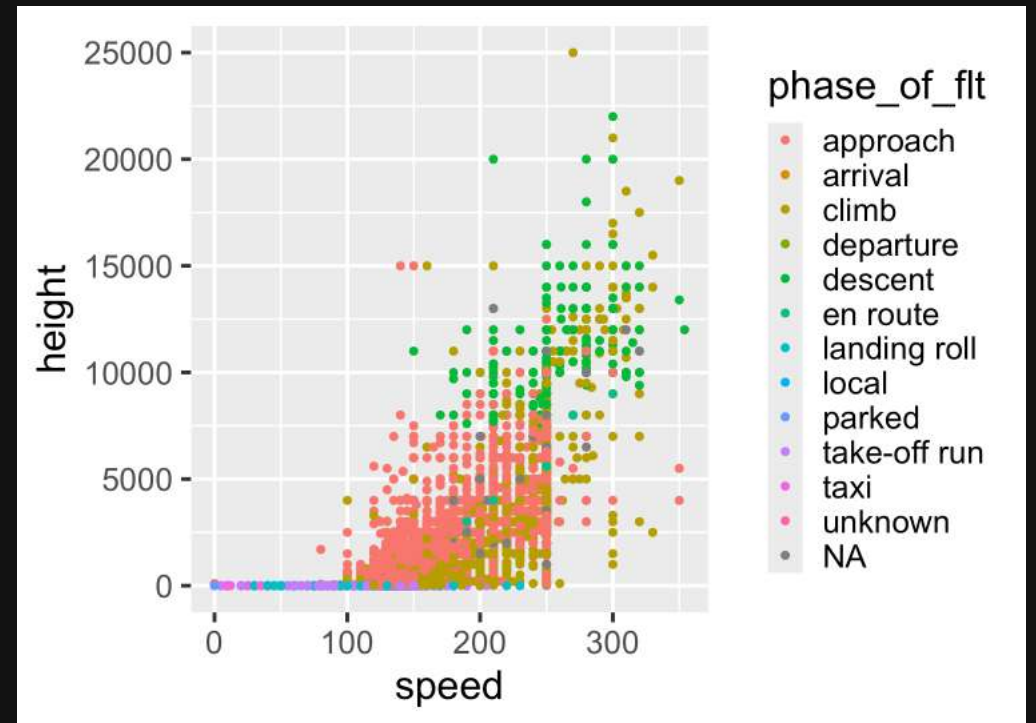
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point(aes(color = gender)) +  
  labs(  
    x = "Year",  
    y = "Age",  
    title = "Bear victim age over time",  
    color = "Gender"  
  )
```



# Your turn: `geom_point()`

10:00

Use the `birds` data frame to create the following plots



*Break*

05:00

# Week 10: *Data Visualization*

1. Plotting with Base R

2. Plotting with ggplot2: Part 1

BREAK

3. Plotting with ggplot2: Part 2

4. Tweaking your ggplot

# Make bar charts with `geom_col()`

With bar charts, you'll often need to create summary variables to plot

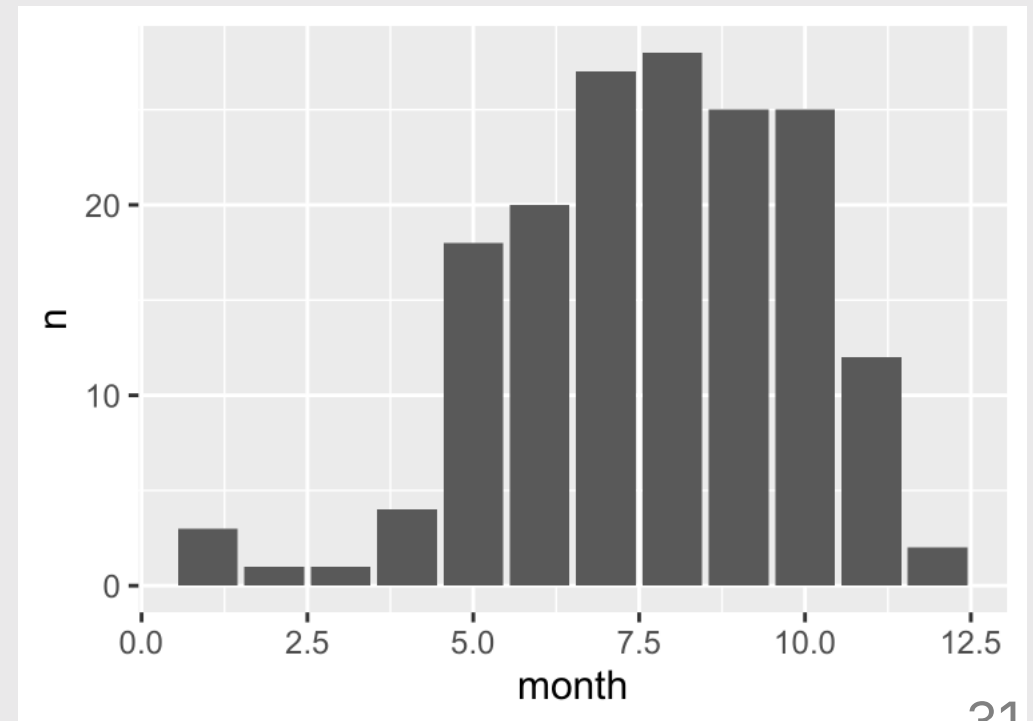
Step 1: Summarize the data

```
bear_months <- bears %>%  
  count(month)
```

Step 2: Make the plot

```
ggplot(data = bear_months) +  
  geom_col(aes(x = month, y = n))
```

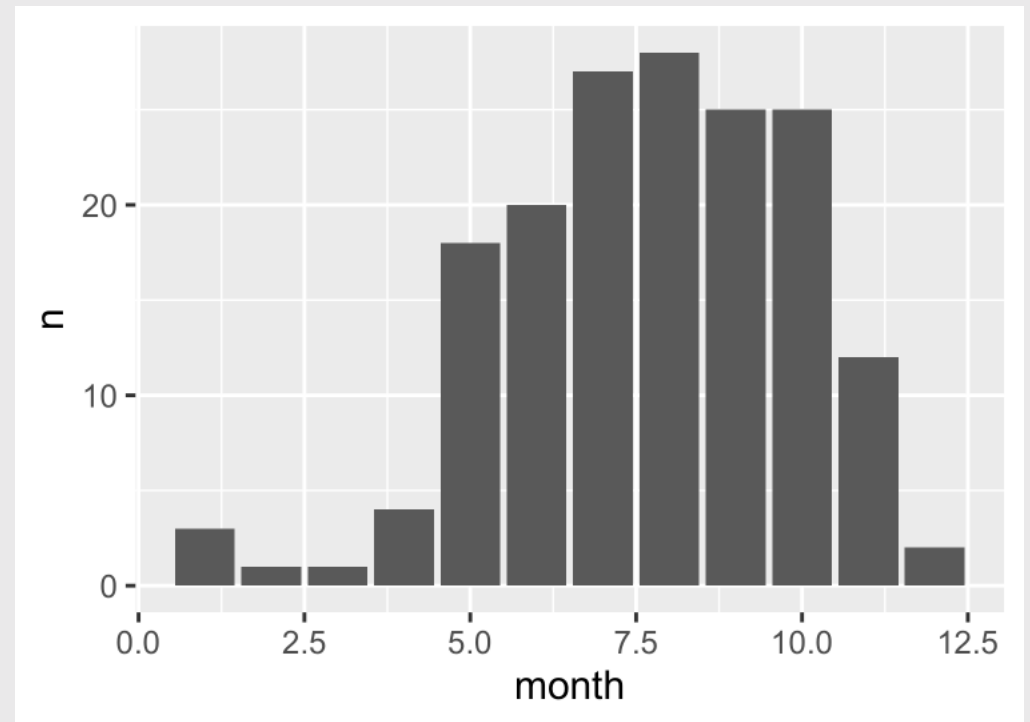
Example: count of attacks by month



# Make bar charts with `geom_col()`

Alternative approach: piping directly into ggplot

```
bears %>%  
  count(month) %>% # Pipe into ggplot  
  ggplot() +  
  geom_col(aes(x = month, y = n))
```



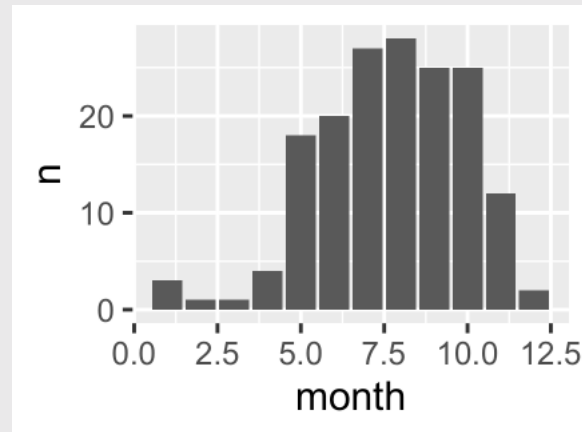


# Be careful with `geom_col()` vs. `geom_bar()`

## `geom_col()`

Map both `x` and `y`

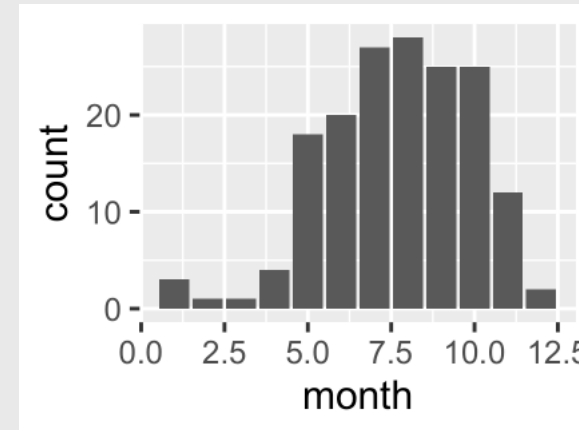
```
bears %>%  
  count(month) %>%  
  ggplot() +  
  geom_col(aes(x = month, y = n))
```



## `geom_bar()`

Only map `x` (`y` is computed)

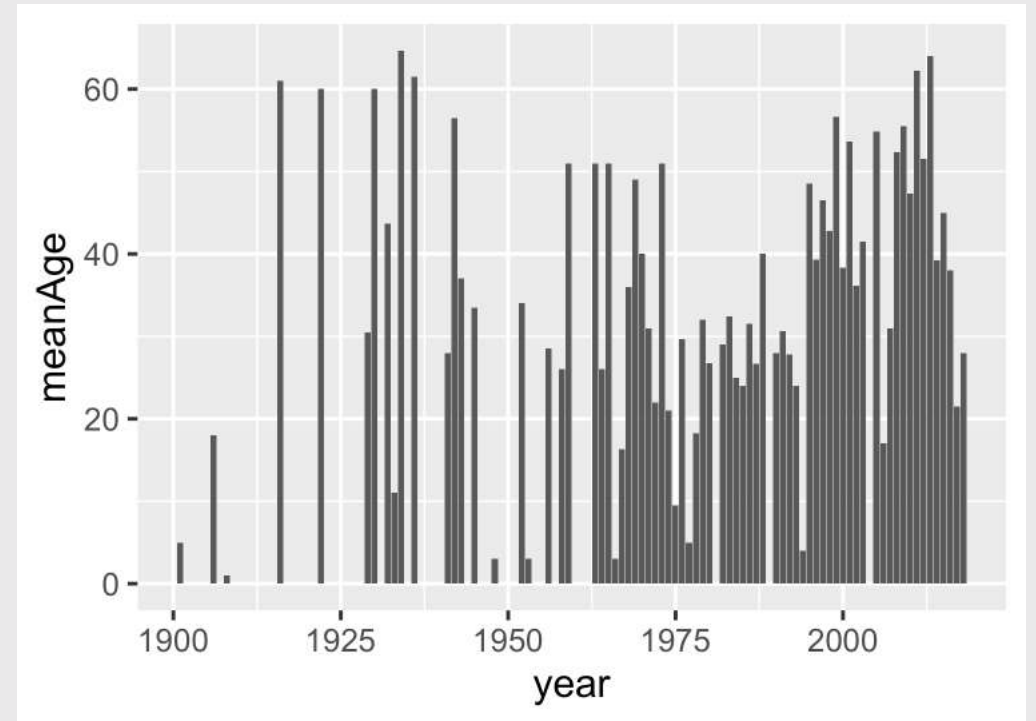
```
bears %>%  
  ggplot() +  
  geom_bar(aes(x = month))
```



# Make bar charts with `geom_col()`

Another example:  
Mean age of victim in each year

```
bears %>%  
  filter(!is.na(age)) %>%  
  group_by(year) %>%  
  summarise(meanAge = mean(age)) %>%  
  ggplot() +  
  geom_col(aes(x = year, y = meanAge))
```

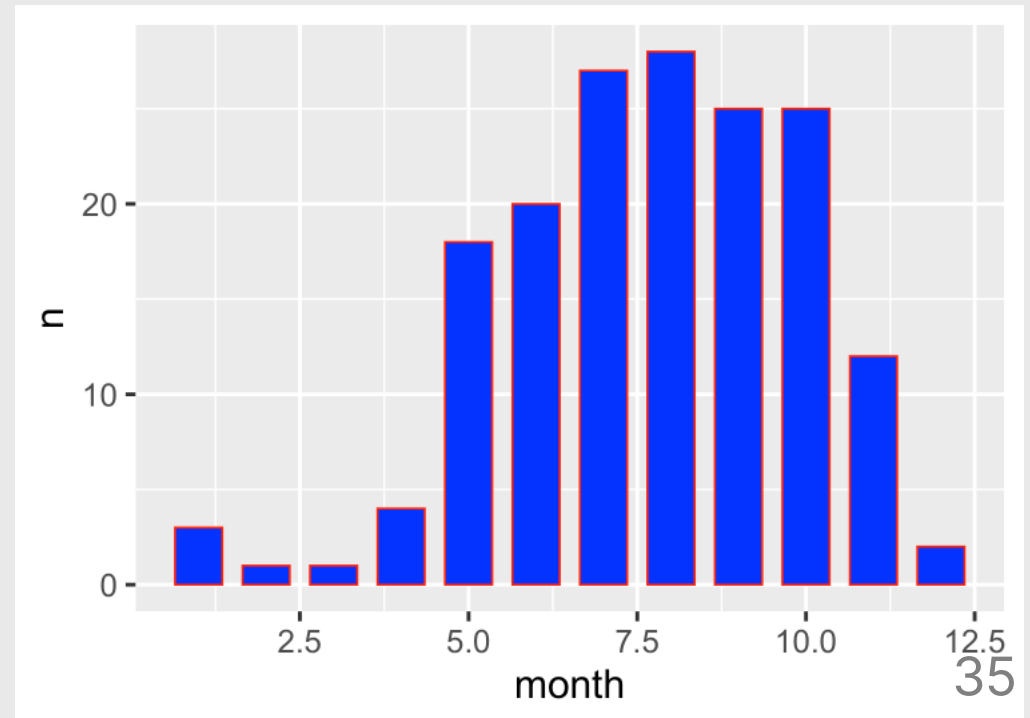


Change bar width: `width`

Change bar color: `fill`

Change bar outline: `color`

```
bears %>%  
  count(month) %>%  
  ggplot() +  
  geom_col(  
    mapping = aes(x = month, y = n),  
    width = 0.7,  
    fill = "blue",  
    color = "red"  
  )
```



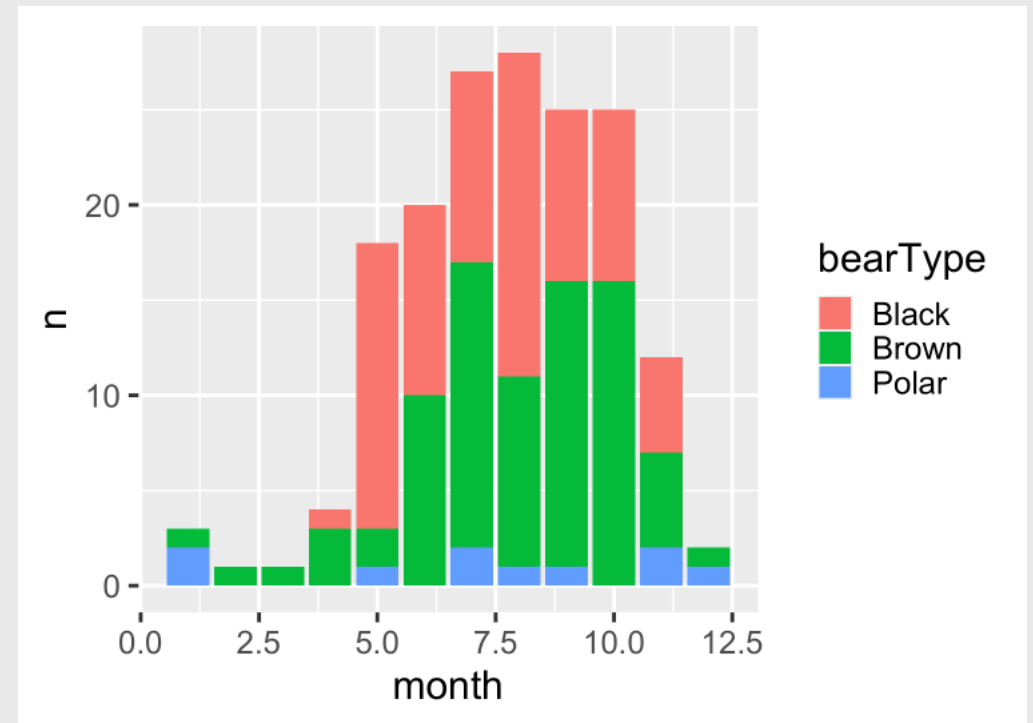
# Map the `fill` to `bearType`

```
bears %>%  
  count(month, bearType) %>%  
  ggplot() +  
  geom_col(  
    mapping = aes(  
      x = month, y = n, fill = bearType)  
    )
```

Note that I had to summarize the count by both `month` and `bearType`

```
bears %>%  
  count(month, bearType)
```

```
#> # A tibble: 27 × 3  
#>   month bearType     n  
#>   <dbl> <chr>      <int>  
#> 1     1 Brown         1  
#> 2     1 Polar         2  
#> 3     2 Brown         1  
#> 4     3 Brown         1  
#> 5     4 Black          1  
#> 6     4 Brown         3
```

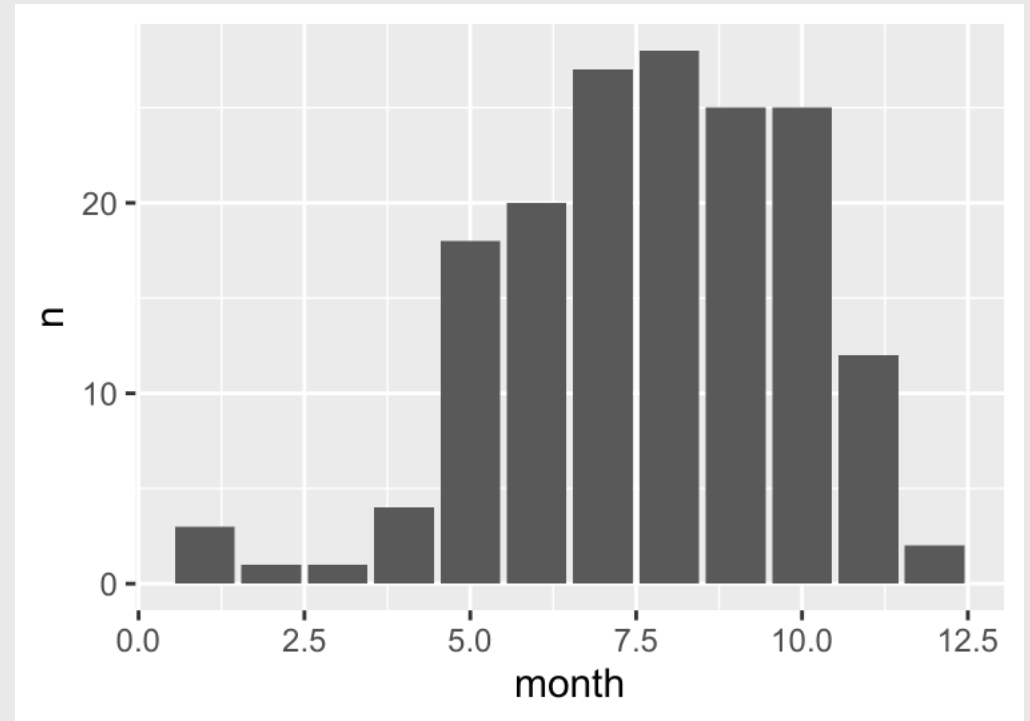


# "Factors" = Categorical variables

By default, R makes numeric variables *continuous*

```
bears %>%  
  count(month) %>%  
  ggplot() +  
  geom_col(aes(x = month, y = n))
```

**The variable `month` is a *number***

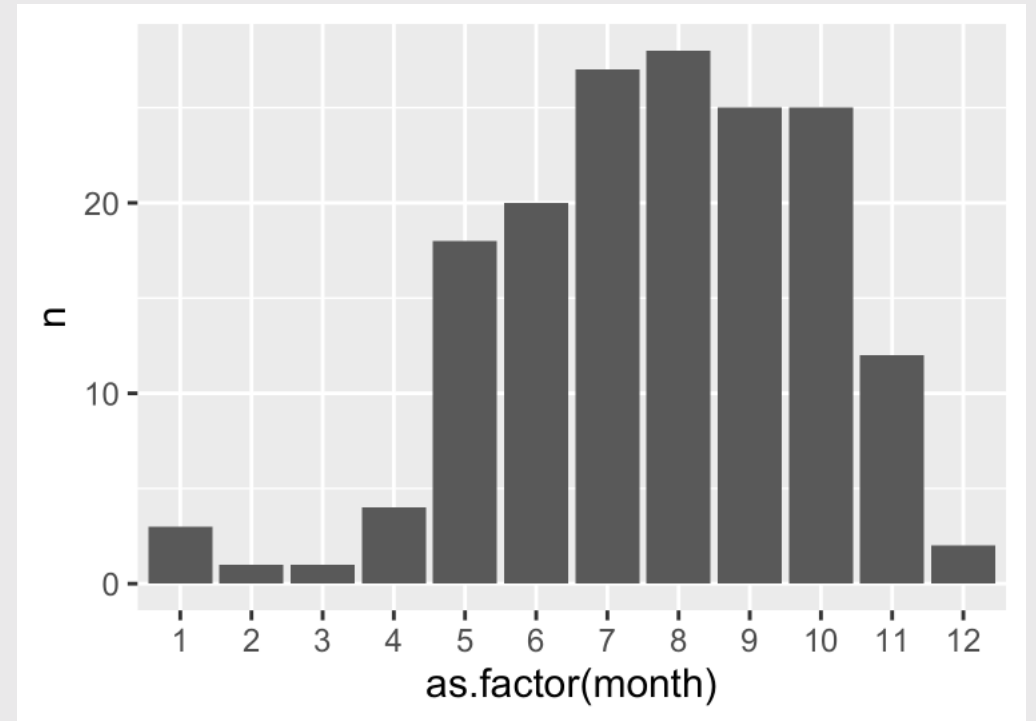


# "Factors" = Categorical variables

You can make a continuous variable *categorical* using `as.factor()`

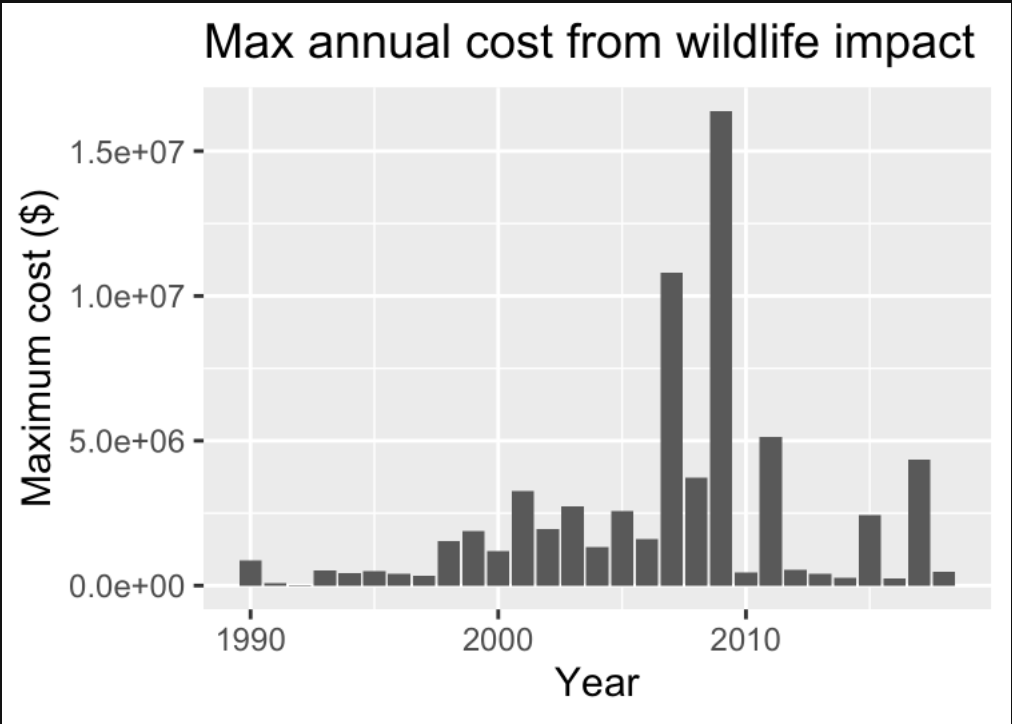
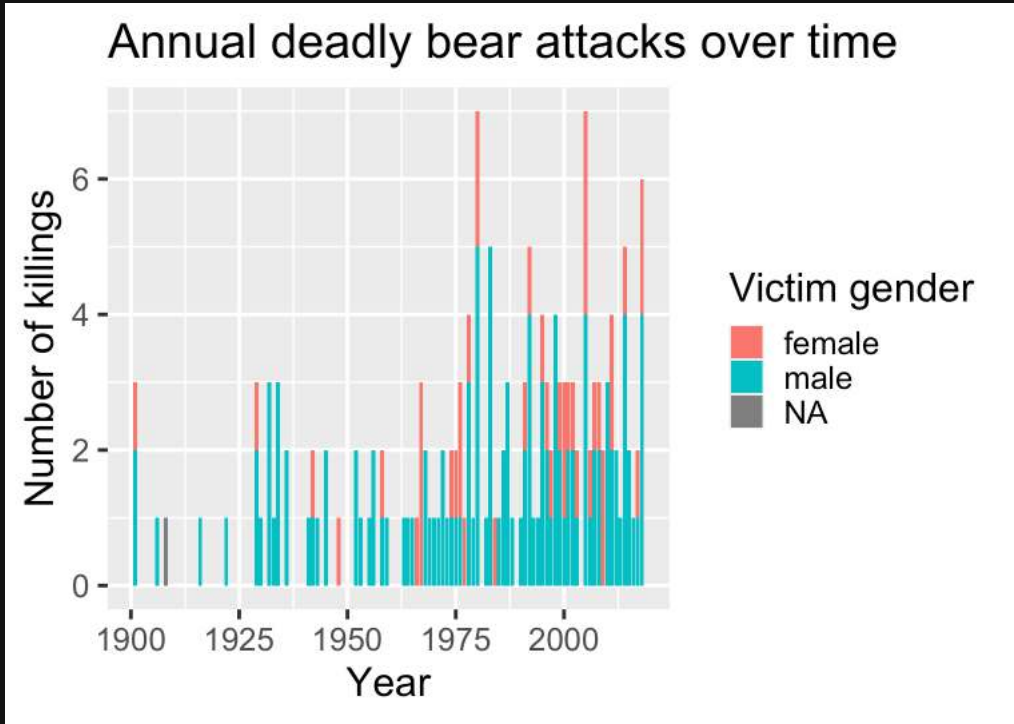
```
bears %>%  
  count(month) %>%  
  ggplot() +  
  geom_col(  
    mapping = aes(  
      x = as.factor(month),  
      y = n)  
    )
```

**The variable `month` is a *factor***



# Your turn: `geom_col()`

Use the `bears` and `birds` data frame to create the following plots



# Week 10: *Data Visualization*

1. Plotting with Base R

2. Plotting with ggplot2: Part 1

BREAK

3. Plotting with ggplot2: Part 2

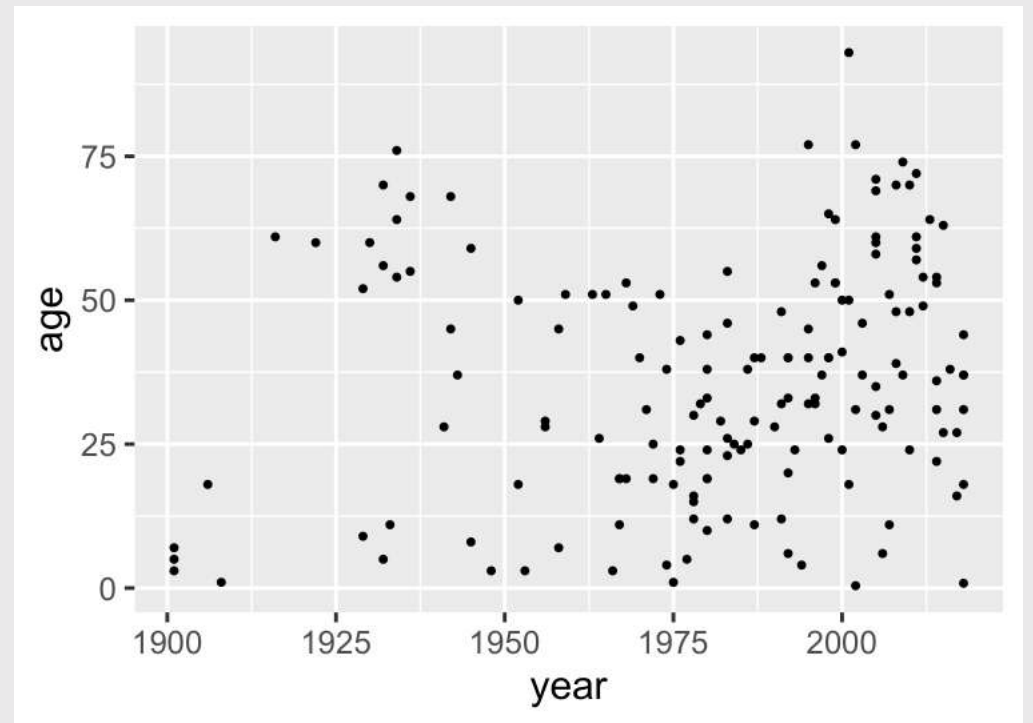
4. Tweaking your ggplot



# Working with themes

Themes change *global* features of your plot, like the background color, grid lines, etc.

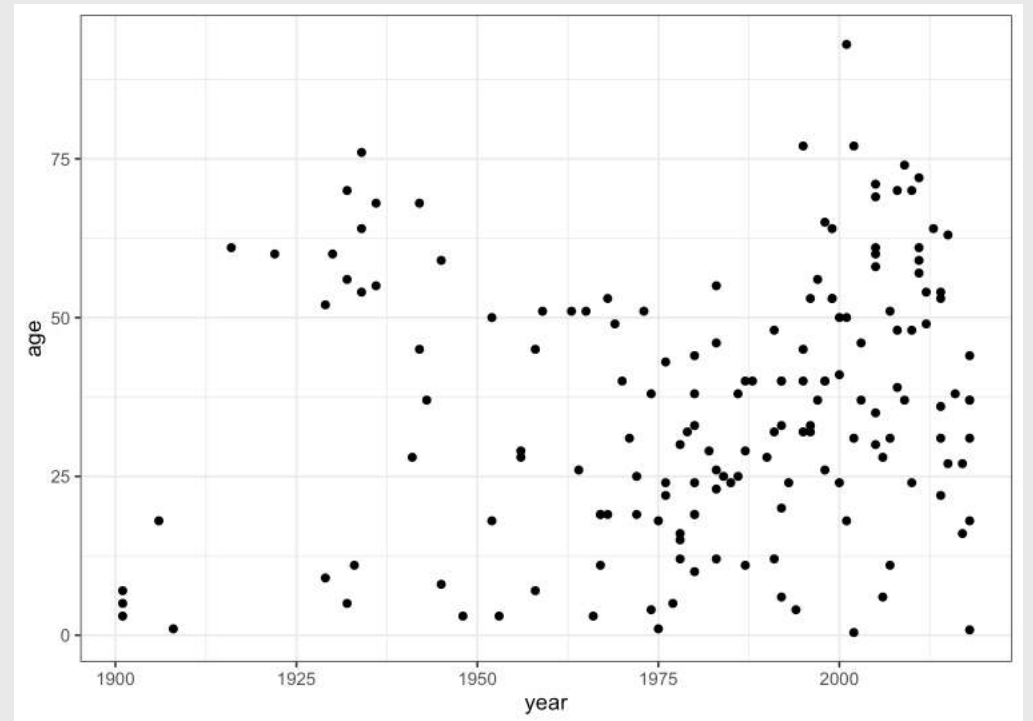
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point()
```



# Working with themes

Themes change *global* features of your plot, like the background color, grid lines, etc.

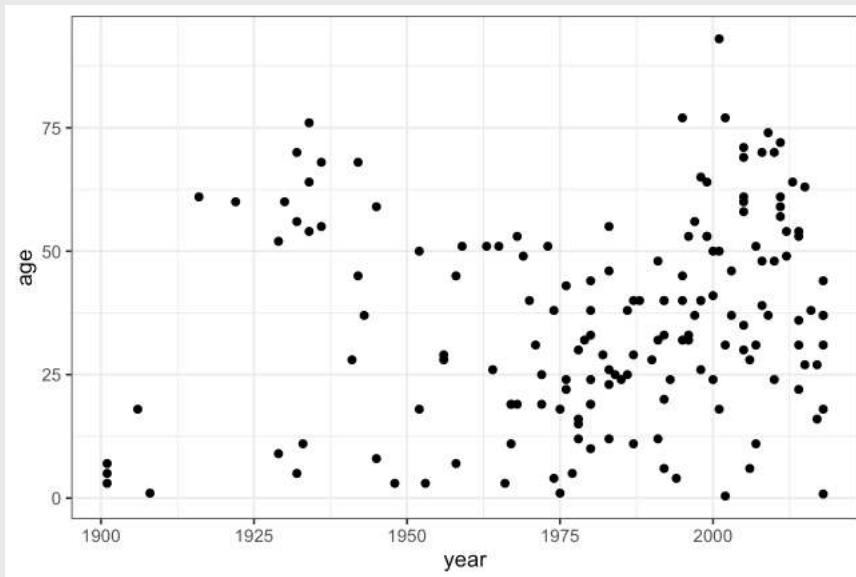
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_bw()
```



# Common themes

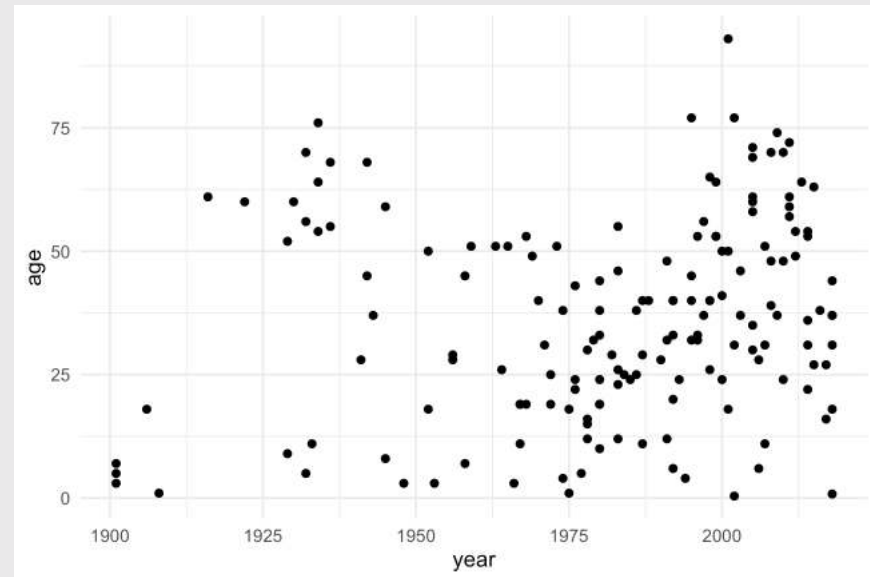
## theme\_bw()

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_bw()
```



## theme\_minimal()

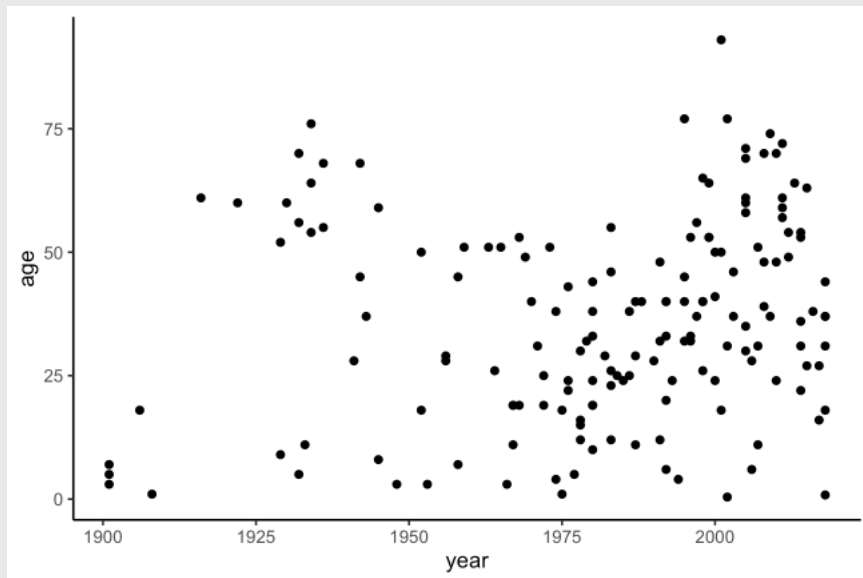
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_minimal()
```



# Common themes

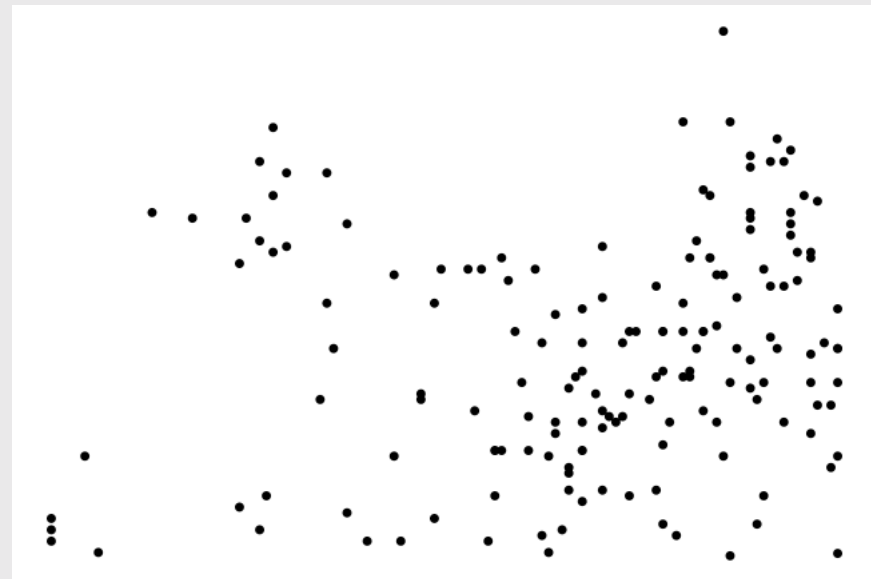
## theme\_classic()

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_classic()
```



## theme\_void()

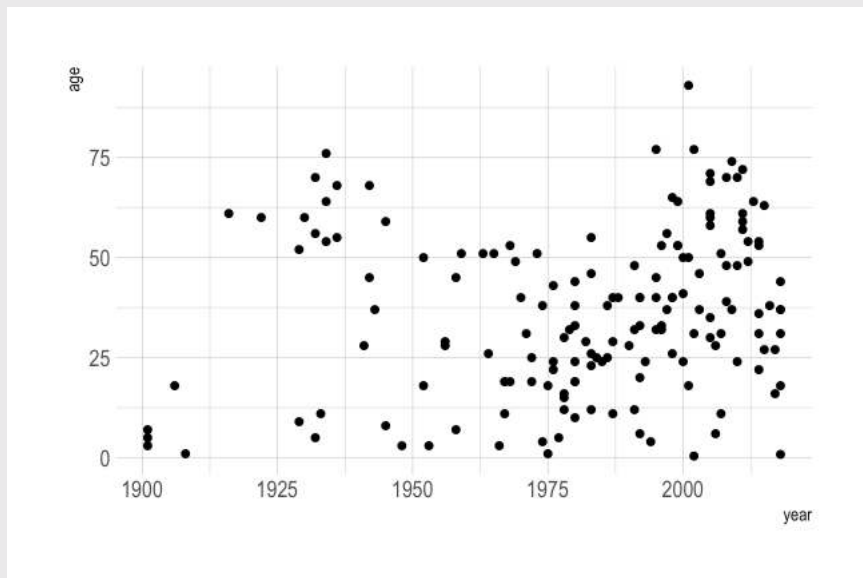
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_void()
```



## Other themes: [hrbrthemes](#)

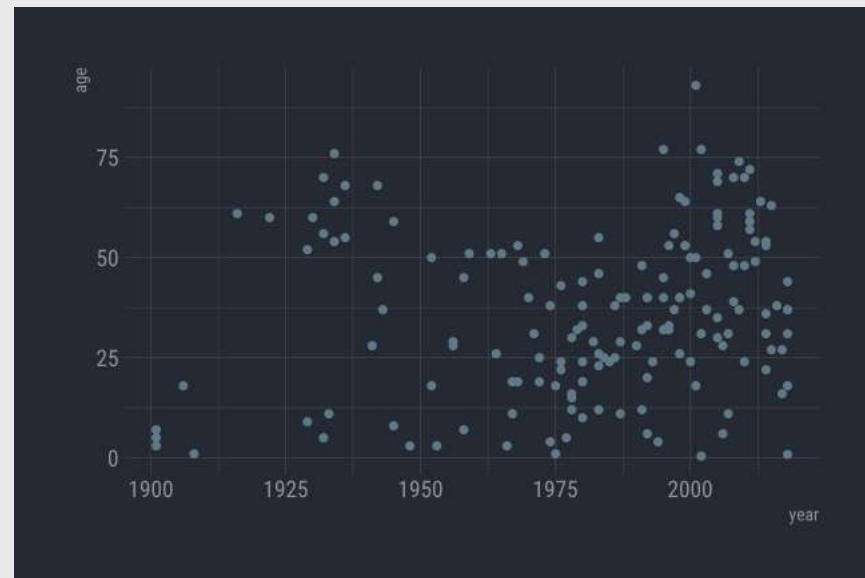
```
library(hrbrthemes)
```

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_ipsum()
```



```
library(hrbrthemes)
```

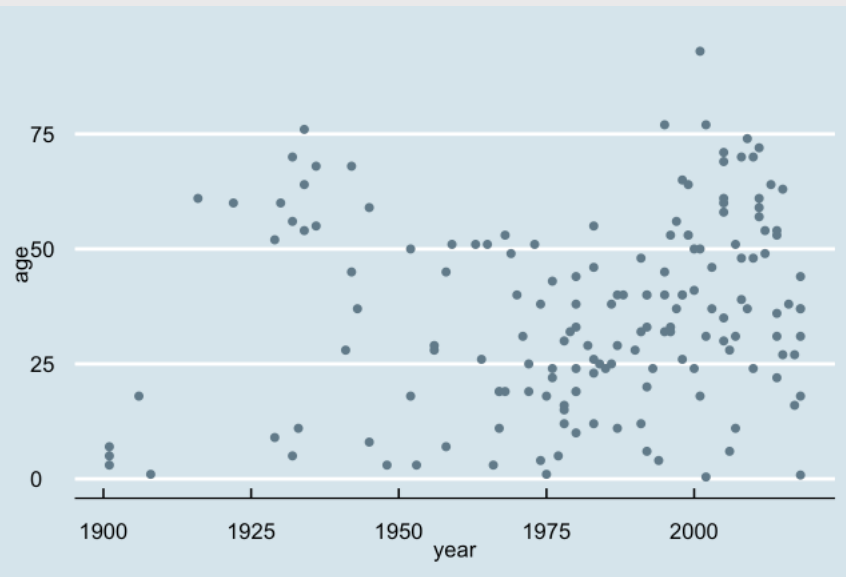
```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_ft_rc()
```



# Other themes: **ggthemes**

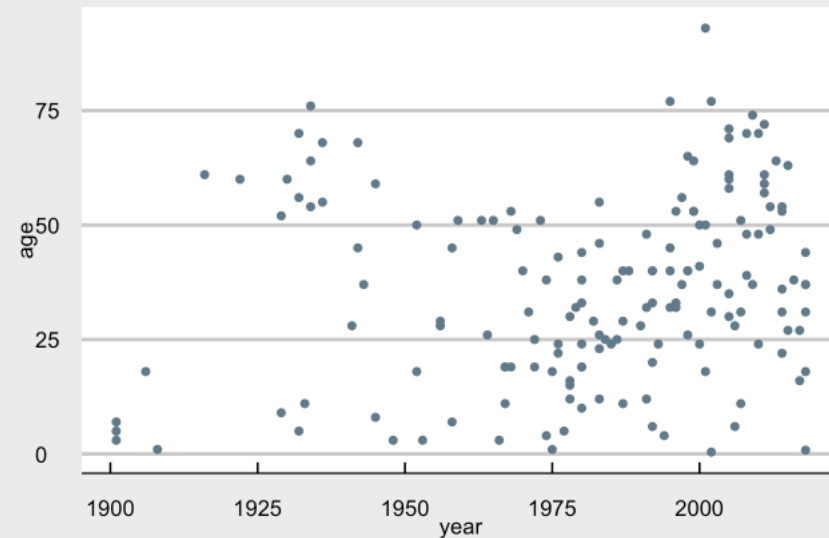
```
library(ggthemes)
```

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_economist()
```



```
library(ggthemes)
```

```
ggplot(  
  data = bears,  
  mapping = aes(x = year, y = age)) +  
  geom_point() +  
  theme_economist_white()
```



# Save figures with `ggsave()`

First, assign the plot to an object name:

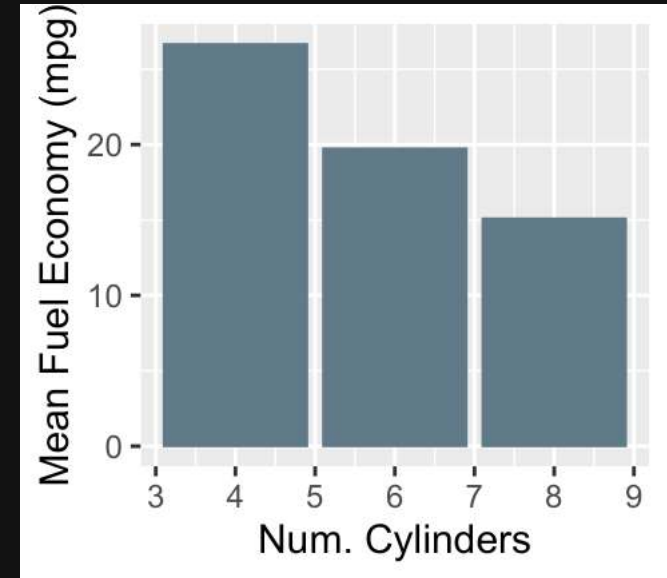
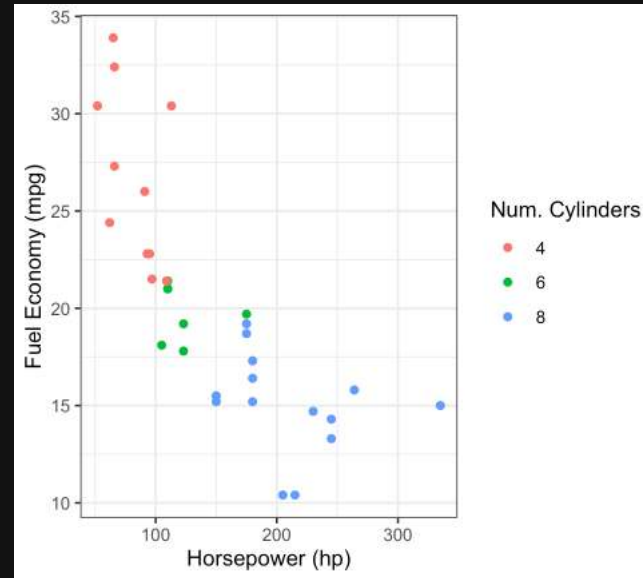
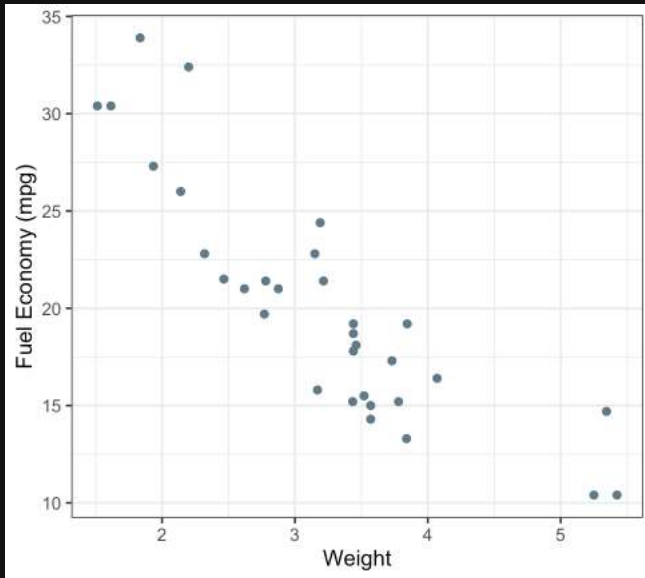
```
scatterPlot <- ggplot(data = bears) +  
  geom_point(aes(x = year, y = age))
```

Then use `ggsave()` to save the plot:

```
ggsave(  
  filename = here('plots', 'scatterPlot.png'),  
  plot      = scatterPlot,  
  width     = 6, # inches  
  height    = 4  
)
```

# Extra practice 1

Use the `mtcars` data frame to create the following plots





# Extra practice 2

Use the `mpg` data frame to create the following plot

