Loading Libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.4     v purrr   0.3.4
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(rvest)
```

```
##
## Attaching package: 'rvest'

## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
library(magrittr)
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 4.1.2
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
##
## Attaching package: 'ggmap'
```

```
## The following object is masked from 'package:magrittr':
##
##     inset
```

```
library(stringr)
```

Loading Dataset

```
atheletes <- read.csv("./Olympics/athlete_events.csv", stringsAsFactors = F)
regions <- read.csv("./Olympics/noc_regions.csv", stringsAsFactors = F)
```

```
df <- atheletes %>%
  group_by(Season, Sex) %>%
  summarise(Count = n()) %>%
  mutate(Percentage = round(Count*100 / sum(Count)))
```

```
## `summarise()` has grouped output by 'Season'. You can override using the `.groups` argument.
```

```
atheletes %>%
  group_by(Year, Season) %>%
  summarise(NumberOfParticipants = n())
```

```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```

```
## # A tibble: 51 x 3
## # Groups:   Year [35]
##     Year Season NumberOfParticipants
##    <int> <chr>                 <int>
##  1  1896 Summer                  380
##  2  1900 Summer                 1936
##  3  1904 Summer                 1301
##  4  1906 Summer                 1733
##  5  1908 Summer                 3101
##  6  1912 Summer                 4040
##  7  1920 Summer                 4292
##  8  1924 Summer                 5233
##  9  1924 Winter                  460
## 10  1928 Summer                 4992
## # ... with 41 more rows
```

```
atheletes$Age[is.na(atheletes$Age)] <- median(atheletes$Age, na.rm = T)
```

```
atheletes <- atheletes %>%
              left_join(regions, by = "NOC")
```

```
Count <- atheletes %>%
  group_by(Year, Season, region) %>%
  summarise(NumberOfAthltes = n())
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
Gold_Winners <- atheletes %>%
                filter(Medal != "<NA>")%>%
                group_by(Year, Season, region) %>%
                summarise(NumberOfMedals = n())
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
Aggregated <- Count %>% left_join(Gold_Winners, by = c("Year", "Season", "region"))

groupMale <- atheletes %>%
              filter(Sex == "M") %>%
              group_by(Year, Season, region) %>%
              summarise(Number_Of_Men = n())
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
groupFemale <- atheletes %>%
              filter(Sex == "F") %>%
              group_by(Year, Season, region) %>%
              summarise(Number_Of_Women = n())
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
group <- groupMale %>%
          left_join(groupFemale) %>%
          mutate(Sex_Ratio = Number_Of_Men/Number_Of_Women)
```

```
## Joining, by = c("Year", "Season", "region")
```

```
group$Sex_Ratio[is.na(group$Sex_Ratio)] <- 236

Aggregated <- Aggregated %>%
              left_join(group, by = c("Year", "Season", "region"))

AgeAgg <- atheletes %>%
                group_by(Year, Season, region) %>%
                summarise(MedianAge = median(Age, na.rm = T))
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
HeightAgg <- atheletes %>%
                group_by(Year, Season, region) %>%
                summarise(MedianHeight = median(Height, na.rm = T))
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
WeightAgg <- atheletes %>%
                group_by(Year, Season, region) %>%
                summarise(MedianWeight = median(Weight, na.rm = T))
```

```
## `summarise()` has grouped output by 'Year', 'Season'. You can override using the `.groups` argument.
```

```
Aggregated <- Aggregated %>%
              left_join(AgeAgg, by = c("Year", "Season", "region"))
Aggregated <- Aggregated %>%
              left_join(HeightAgg, by = c("Year", "Season", "region"))
Aggregated <- Aggregated %>%
              left_join(WeightAgg, by = c("Year", "Season", "region"))

Aggregated$NumberOfMedals[is.na(Aggregated$NumberOfMedals)] <- 0
Aggregated$Sex_Ratio[is.na(Aggregated$Sex_Ratio)] <- 0
```

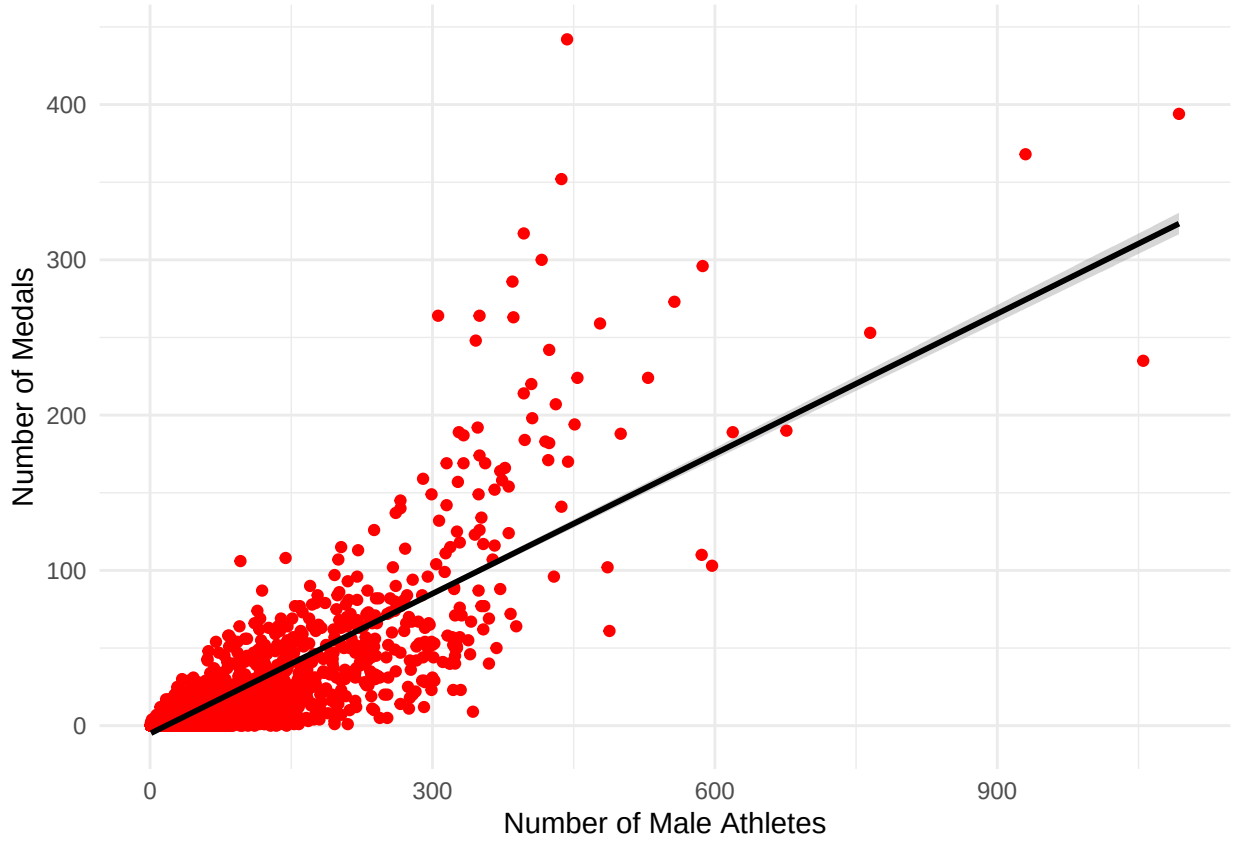
Impact On Medals

```


Aggregated %>%
        filter(!is.na(Number_Of_Men)) %>%
        ggplot(aes(x=Number_Of_Men, y=NumberOfMedals)) +
        geom_point(col="red")  + geom_smooth(method = "lm", se=TRUE, color="black", aes(group=1)) +
        theme_minimal() +
        labs(x = "Number of Male Athletes", y = "Number of Medals")
```

Number Of Males

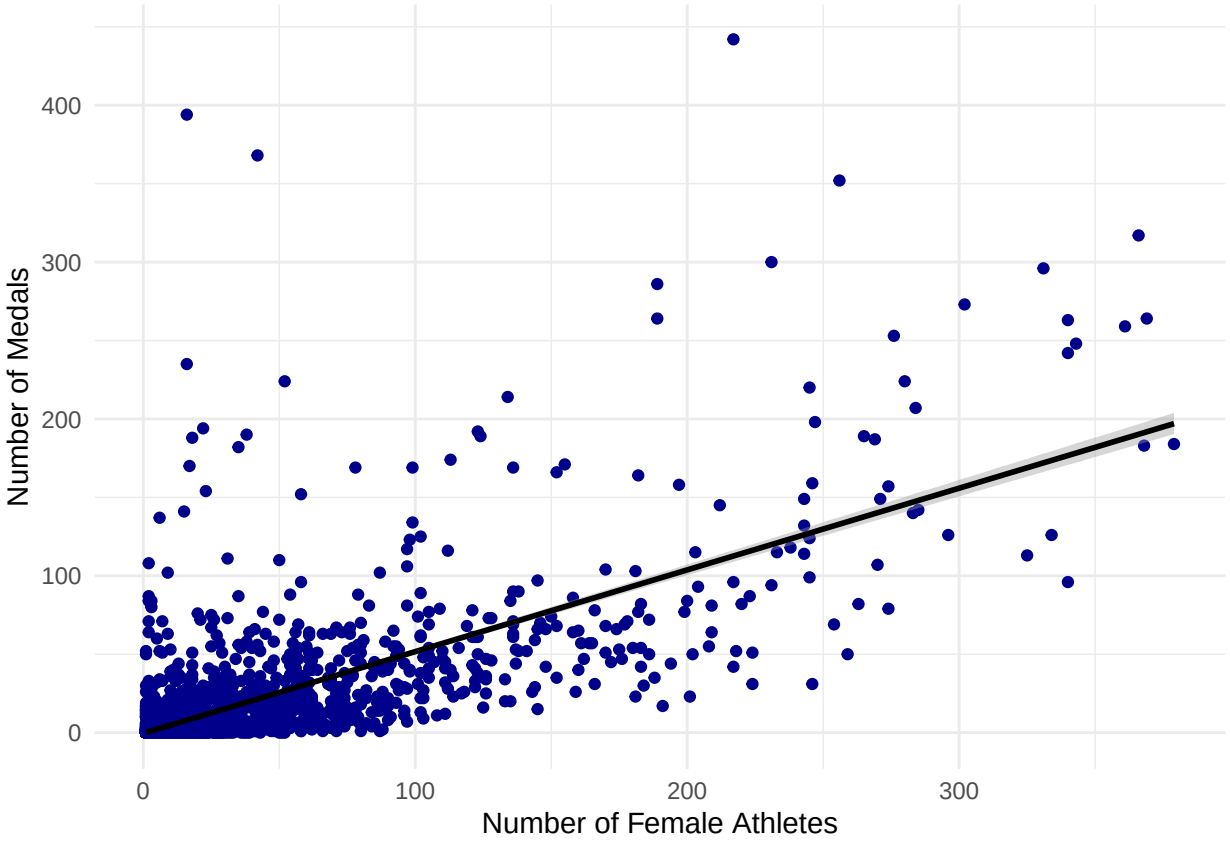
```
## `geom_smooth()` using formula 'y ~ x'
```




```
Aggregated %>%
        filter(!is.na(Number_Of_Women)) %>%
        ggplot(aes(x=Number_Of_Women, y=NumberOfMedals)) +
        geom_point(col="darkblue")  + geom_smooth(method = "lm", se=TRUE, color="black", aes(group=1)) +
        theme_minimal() +
        labs(x = "Number of Female Athletes", y = "Number of Medals")
```

Number Of Females

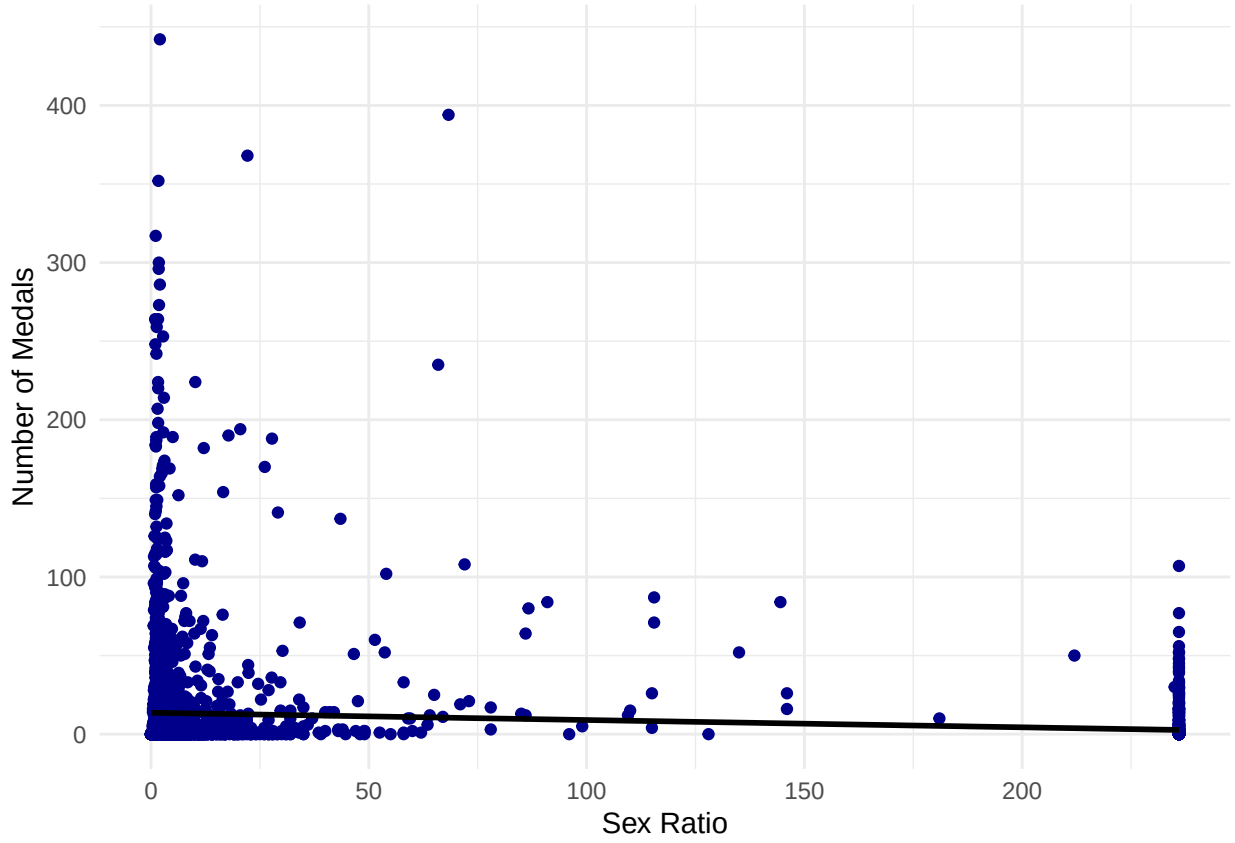
```
## `geom_smooth()` using formula 'y ~ x'
```




```
Aggregated %>%
       filter(!is.na(Sex_Ratio)) %>%
       ggplot(aes(x=Sex_Ratio, y=NumberOfMedals)) +
       geom_point(col="darkblue")  + geom_smooth(method = "lm", se=TRUE, color="black", aes(group=1)) +
       theme_minimal() +
       labs(x = "Sex Ratio", y = "Number of Medals")
```

Sex Ratio

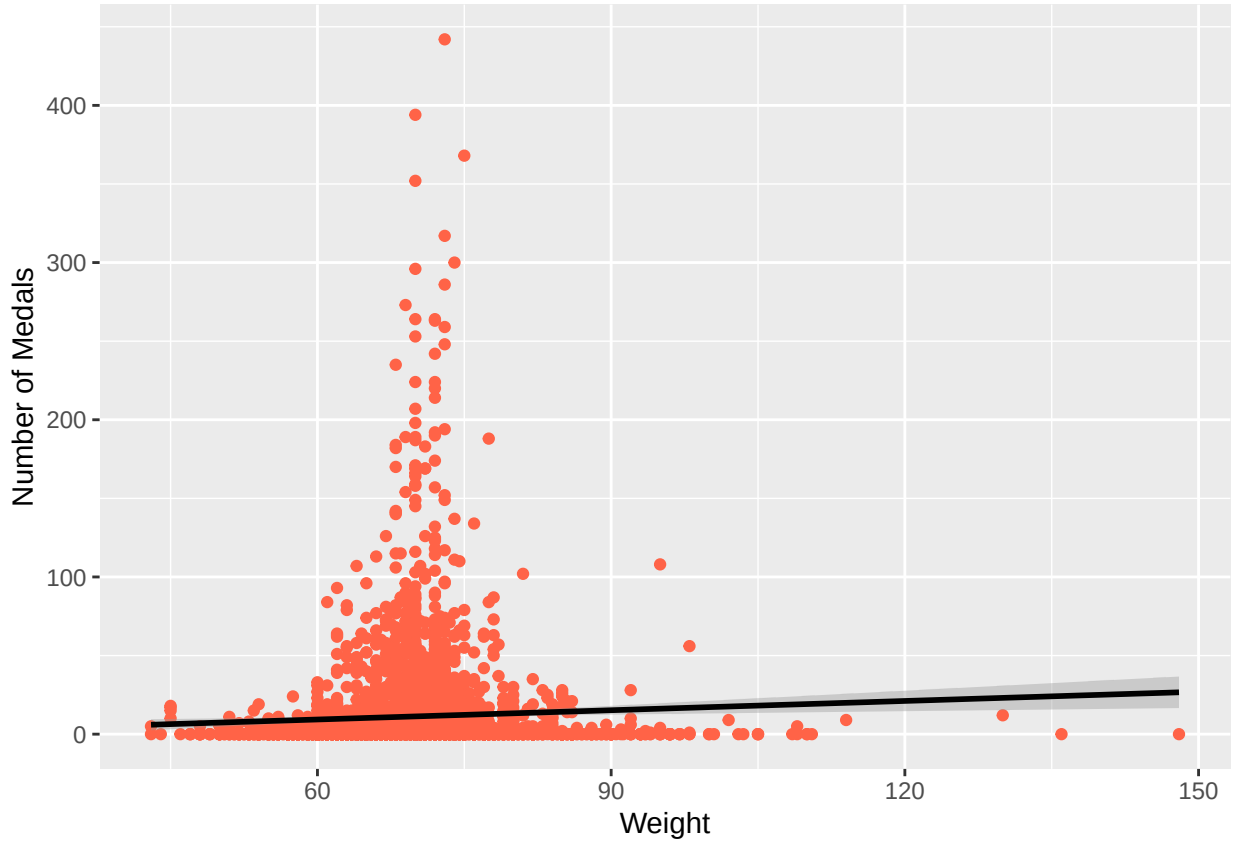
```
## `geom_smooth()` using formula 'y ~ x'
```




```
Aggregated %>%
        filter(!is.na(MedianWeight)) %>%
        ggplot(aes(x=MedianWeight, y=NumberOfMedals)) +
        geom_point(col="tomato")  + geom_smooth(method = "lm", se=TRUE, color="black", aes(group=1)) +
        labs(x = "Weight", y = "Number of Medals")
```

Weight

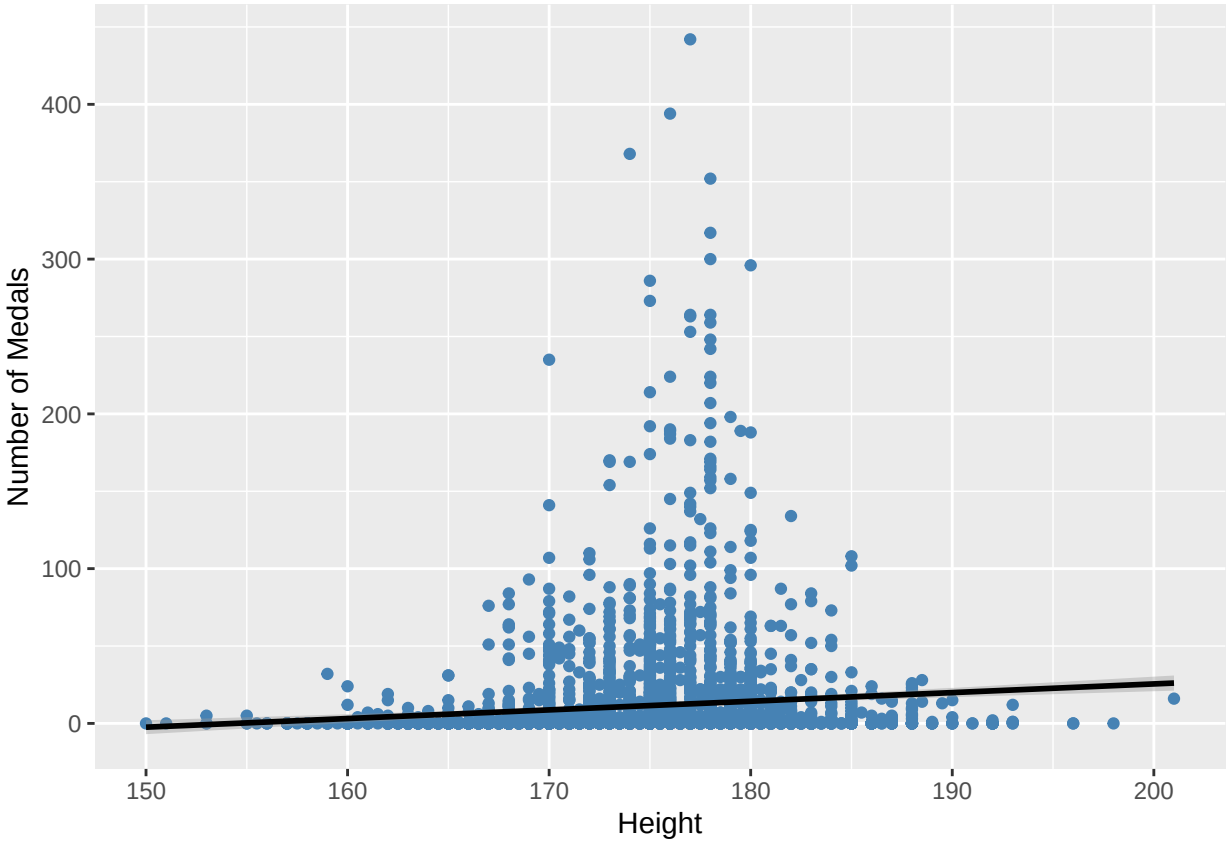
```
## `geom_smooth()` using formula 'y ~ x'
```




```
Aggregated %>%
        filter(!is.na(MedianHeight)) %>%
        ggplot(aes(x=MedianHeight, y=NumberOfMedals)) +
        geom_point(col="steelblue")  + geom_smooth(method = "lm", se=TRUE, color="black", aes(group=1))
        labs(x = "Height", y = "Number of Medals")
```

Height

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


Number of Medals
400
300
200
100
0
150
160
170
180
190
200
Height