Pokemon Plot

Juan Machuca

8/25/2020

Plotting Pokemon Data using Tidyverse and some of its packages

This is an example on how to make simple plots using ggplot2, dplyr and other tidyverse libraries. The data is extracted directly from the web from Keith Galli's Github Pokemon Repository. In this brief work, I make a quick overview of R syntax to make simple bar charts for the purpose of exploratory data analysis and data visualization. I also employed the dplyr package for the purpose of data wrangling and manipulation.

```
#Download tidyverse library and pull data from github
library(tidyverse)
```

```
## -- Attaching packages -------------------------------------------------------------------------
---------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

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

```
pokedex <- read.csv("https://raw.githubusercontent.com/KeithGalli/pandas/master/pokemon_data.cs
v")
head(pokedex)
```

```
##   X.                  Name Type.1 Type.2 HP Attack Defense Sp..Atk Sp..Def
## 1  1             Bulbasaur  Grass Poison 45     49      49      65      65
## 2  2               Ivysaur  Grass Poison 60     62      63      80      80
## 3  3              Venusaur  Grass Poison 80     82      83     100     100
## 4  3 VenusaurMega Venusaur  Grass Poison 80    100     123     122     120
## 5  4            Charmander   Fire        39     52      43      60      50
## 6  5            Charmeleon   Fire        58     64      58      80      65
##   Speed Generation Legendary
## 1    45          1     FALSE
## 2    60          1     FALSE
## 3    80          1     FALSE
## 4    80          1     FALSE
## 5    65          1     FALSE
## 6    80          1     FALSE
```

```
# Use dplyr package for:
#counting pokemon by type 1 and both type 1 & 2

#Count Type 1
pokemons_by_type <- pokedex %>% group_by (Type.1) %>% summarise(pokemons_type=n())%>% arrange(de
sc(pokemons_type))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

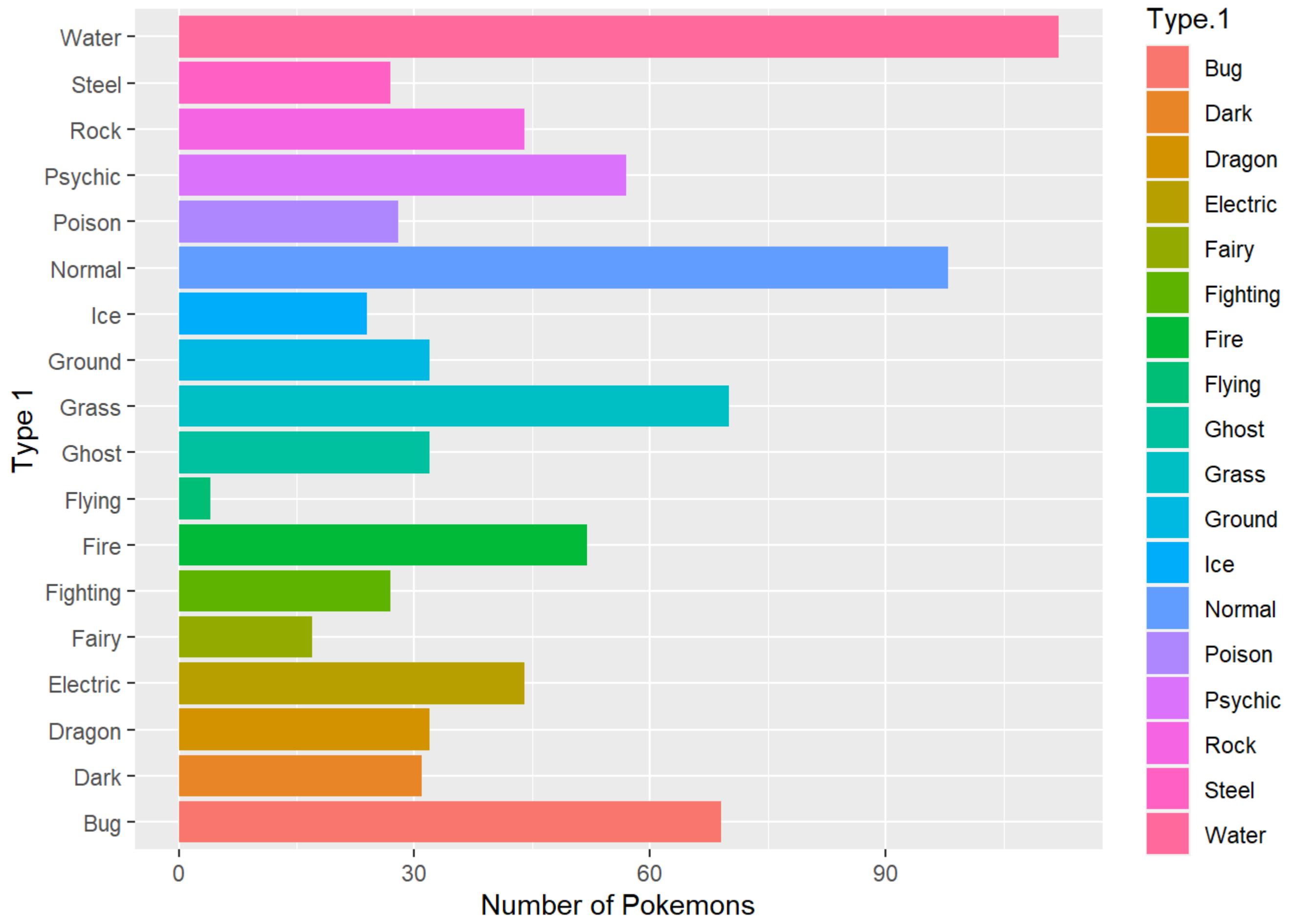
```
pokemons_by_type <- as.data.frame(pokemons_by_type)

#Count Type 1 & 2
pokemons_type_total <- pokedex %>% select(Type.1, Type.2) %>%  gather("Types")
pokemon_type_total <- pokemons_type_total %>% group_by(value) %>% filter(value != "") %>% summar
ise(count=n())%>% arrange(desc(count))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

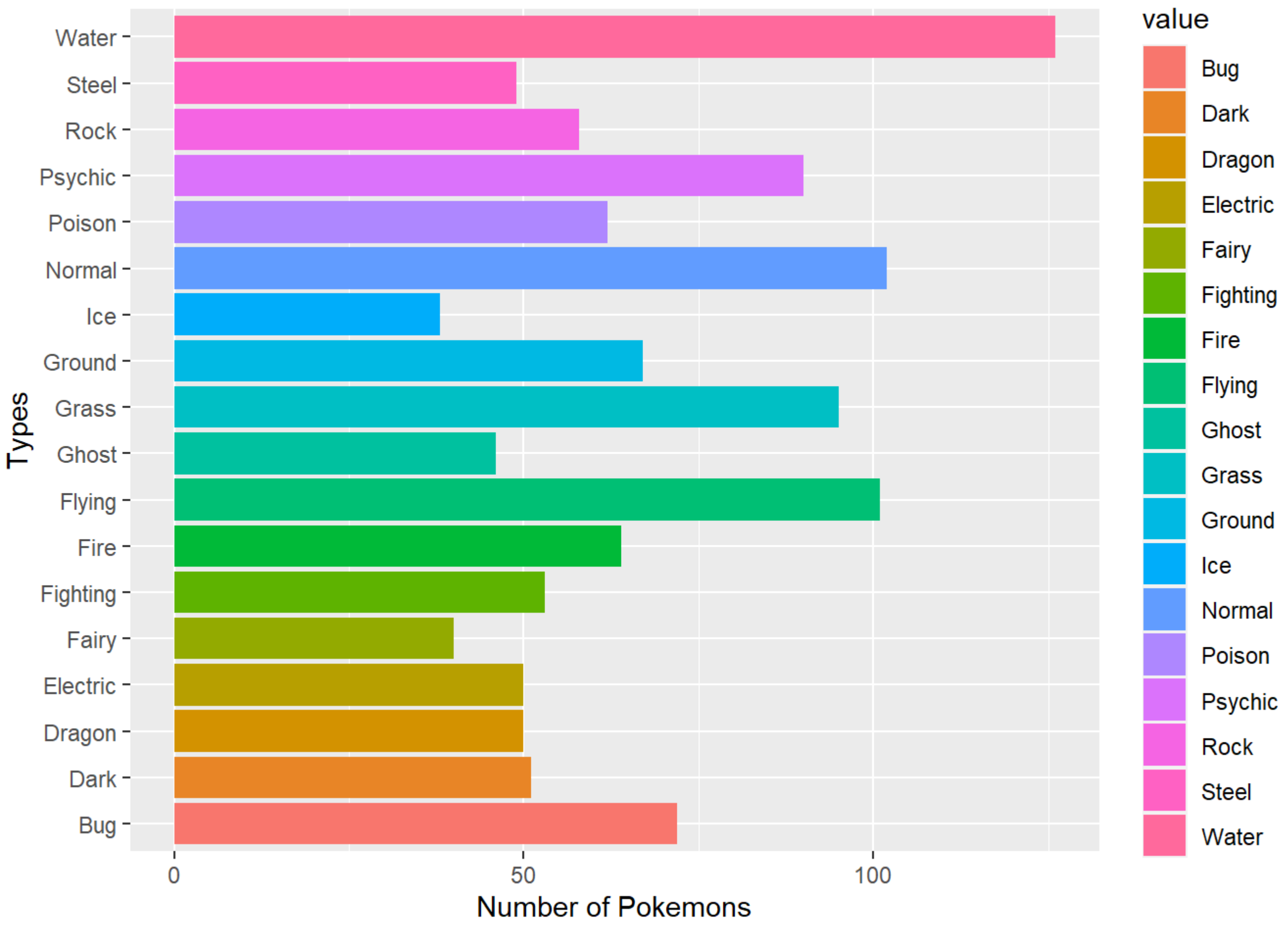
```
pokemon_type_total
```

```
## # A tibble: 18 x 2
##    value    count
##    <chr>    <int>
##  1 Water      126
##  2 Normal     102
##  3 Flying     101
##  4 Grass       95
##  5 Psychic     90
##  6 Bug         72
##  7 Ground      67
##  8 Fire        64
##  9 Poison      62
## 10 Rock        58
## 11 Fighting    53
## 12 Dark        51
## 13 Dragon      50
## 14 Electric    50
## 15 Steel       49
## 16 Ghost       46
## 17 Fairy       40
## 18 Ice         38
```

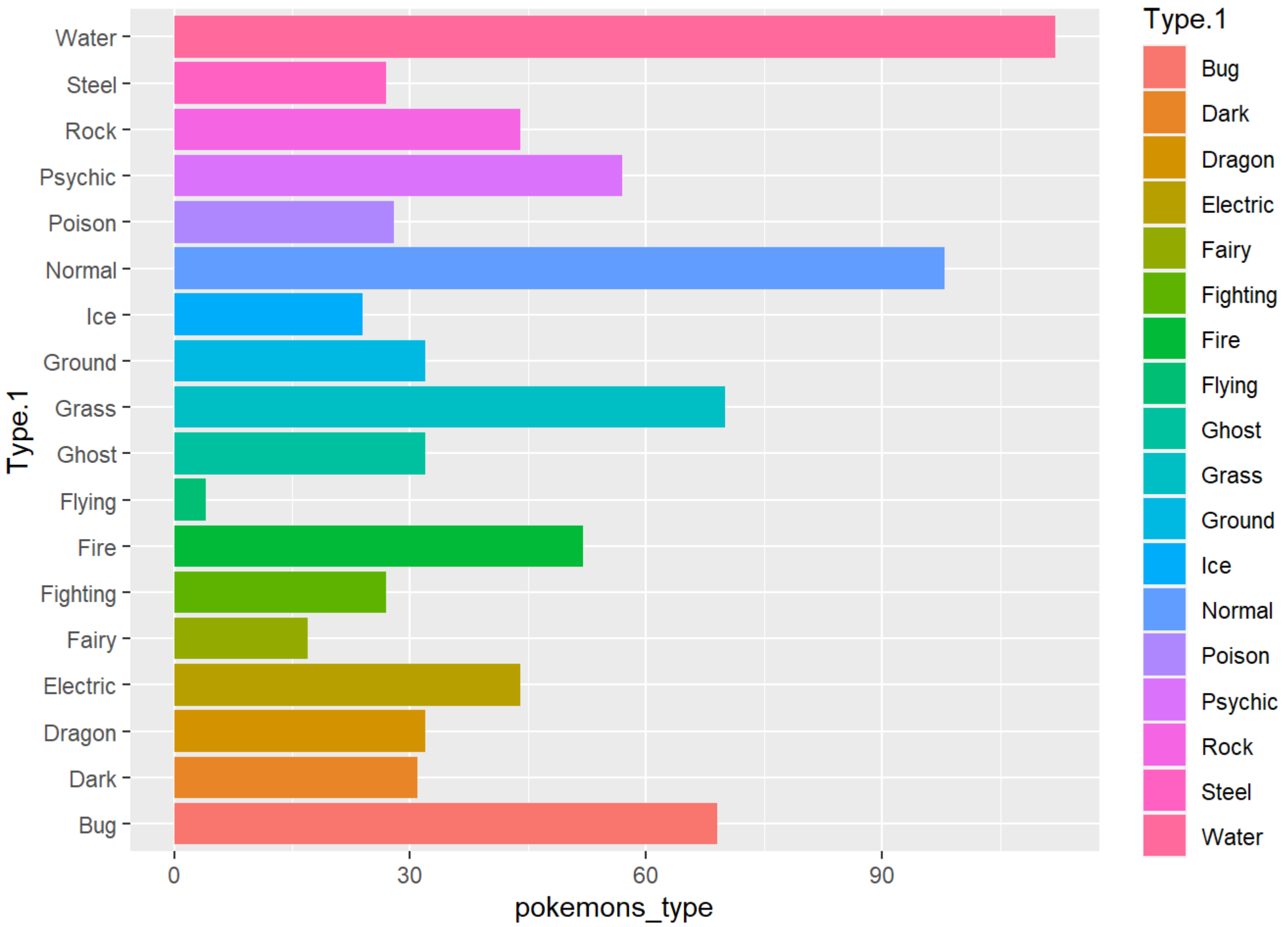
```
#Plot in Base R Pokemon type 1 and total
ggplot(pokemons_by_type, aes(Type.1, pokemons_type, fill = Type.1)) +
  geom_col() +
  coord_flip() +
  ylab("Number of Pokemons") +
  xlab('Type 1')
```



```
ggplot(pokemon_type_total, aes(x=value,y=count, fill = value)) +
  geom_col() +
  coord_flip() +
  ylab("Number of Pokemons") +
  xlab('Types')
```



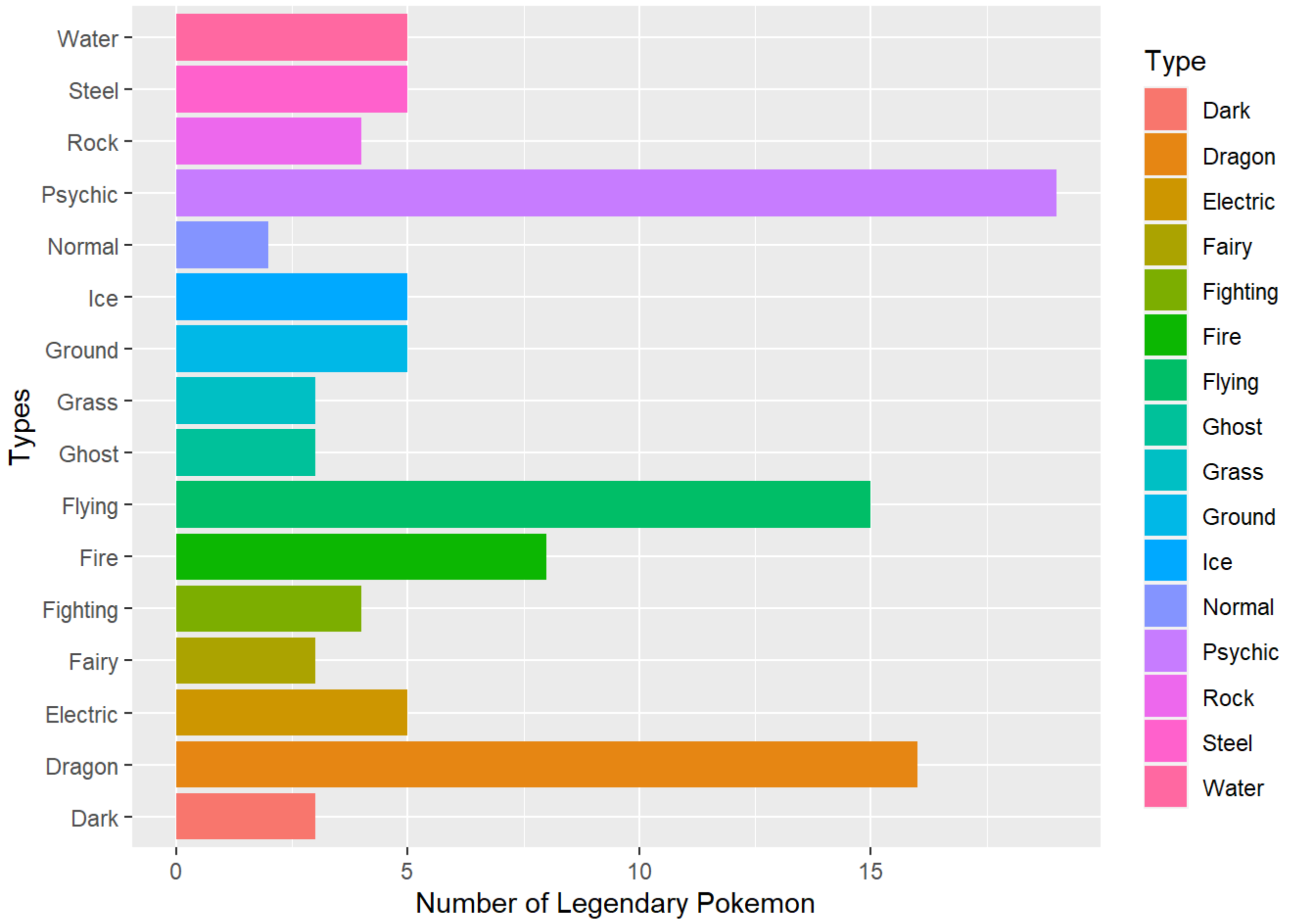
```
#Plot using dplyr
pokemons_by_type %>%
  ggplot()+
  geom_bar(aes(Type.1, pokemons_type, fill = Type.1 ),stat="identity")+
  coord_flip()
```



```
#Legendary Pokemon by Type
legendary <- pokedex %>% select(Name, Type.1, Type.2, Legendary) %>% filter(Legendary==TRUE)
legendary_by_type <-legendary %>% select(Type.1, Type.2, Legendary) %>% gather('Legendary', 'Typ
e')
legendary_by_type <- legendary_by_type %>% group_by(Type) %>% filter(Type!= "") %>% summarise(co
unt=n())%>% arrange(desc(Type))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
view(legendary_by_type)
view(legendary)

#Plot Legendary by Type
legendary_by_type %>%
  ggplot()+
  geom_bar(aes(x=Type, y=count, fill=Type),stat="identity")+
  coord_flip() +
  ylab("Number of Legendary Pokemon") +
  xlab('Types')
```



```
#Summary statistics
summary(pokedex)
```

```
##        X.             Name             Type.1             Type.2
##  Min.   :  1.0   Length:800         Length:800         Length:800
##  1st Qu.:184.8   Class :character   Class :character   Class :character
##  Median :364.5   Mode  :character   Mode  :character   Mode  :character
##  Mean   :362.8
##  3rd Qu.:539.2
##  Max.   :721.0
##        HP             Attack       Defense          Sp..Atk
##  Min.   :  1.00   Min.   :  5   Min.   :  5.00   Min.   : 10.00
##  1st Qu.: 50.00   1st Qu.: 55   1st Qu.: 50.00   1st Qu.: 49.75
##  Median : 65.00   Median : 75   Median : 70.00   Median : 65.00
##  Mean   : 69.26   Mean   : 79   Mean   : 73.84   Mean   : 72.82
##  3rd Qu.: 80.00   3rd Qu.:100   3rd Qu.: 90.00   3rd Qu.: 95.00
##  Max.   :255.00   Max.   :190   Max.   :230.00   Max.   :194.00
##     Sp..Def          Speed          Generation    Legendary
##  Min.   : 20.0   Min.   :  5.00   Min.   :1.000   Mode :logical
##  1st Qu.: 50.0   1st Qu.: 45.00   1st Qu.:2.000   FALSE:735
##  Median : 70.0   Median : 65.00   Median :3.000   TRUE :65
##  Mean   : 71.9   Mean   : 68.28   Mean   :3.324
##  3rd Qu.: 90.0   3rd Qu.: 90.00   3rd Qu.:5.000
##  Max.   :230.0   Max.   :180.00   Max.   :6.000
```

```
summary(legendary_by_type)
```

```
##      Type               count       
##  Length:16          Min.   : 2.000  
##  Class :character   1st Qu.: 3.000  
##  Mode  :character   Median : 5.000  
##                     Mean   : 6.562  
##                     3rd Qu.: 5.750  
##                     Max.   :19.000  
```