Dogs or Cats?

Dash Wieland

3/29/2019

Is that a dog or a cat?

We're going to try and look at the most popular pet names used for dogs and cats in equal numbers.

Start by loading the required packages and downloading the data from GitHub.

```
suppressMessages(library(dplyr)) # for manipulating data
suppressMessages(library(ggplot2)) # for plotting data
suppressMessages(library(forcats)) # for working with categorical variables
```

```
seattle_pets <- readr::read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/da
```

Prepping data for plotting

Dealing with missing values

To start, we need to get rid of rows with missing names or missing species, since those variables are required for our work. First, use piping from dplyr and the filter function to remove rows with missing values in the "animals_name" and "species" columns and create the dataframe "no_missing".

```
no_missing <- seattle_pets %>%
  filter(!is.na(animals_name), !is.na(species))
```

Group by and summarise

Now create an object "grouped_names_species" which groups the "no_missing" dataframe by animal name and species, then counts the occurrences of each value with summarise(n()).

```
grouped_names_species <- no_missing %>%
  group_by(animals_name, species) %>%
  summarise(n())
```

```
head(grouped_names_species) # Take a peek!
```

```
## # A tibble: 6 x 3
## # Groups:   animals_name [6]
##   animals_name                       species `n()`
##   <chr>                              <chr>   <int>
## 1 __                                 Cat         1
## 2 -                                  Cat         1
## 3 'Alani                             Cat         1
## 4 'Murca                             Dog         1
## 5 "\"Luci\" Lucia Rosalin Wicksugal" Dog         1
## 6 "\"Mama\" Maya"                    Cat         1
```

Long to wide with spread()

Getting closer - we have the number of pets with each name, broken out by species. The data is currently in a long format, however, (i.e. "species" is a column with values "Dog" and "Cat", etc. . .), and we want each species as a seperate column. Use the tidyr function spread() to do this.

```
names_species_wide <- tidyr::spread(grouped_names_species, species, `n()`)
```

Filtering down to Cats and Dogs

Since there aren't many pets that aren't dogs or cats, let's filter out the rest of the columns using select() and again remove any missing values using filter and the !is.na() function. Remember the "!" inverts the function.

```
cat_vs_dog_names <- names_species_wide %>%
  select(animals_name, Cat, Dog) %>%
  filter(!is.na(Cat),
         !is.na(Dog))
```

Where cats = dogs using mutate() and filter()

Almost done with data manipulation now. We just need to select the rows where the number of dogs is the same as the number of cats. Since we want the most popular names we'll also select only rows with at least 11 total pets. This will require us to create a "total" column, which is easy with mutate().

```
popular_cats_and_dogs <- cat_vs_dog_names %>%
  mutate(Total = Cat + Dog) %>%
  filter(Cat == Dog,
         Total >=11) %>%
  arrange(desc(Total))
```

Ready to plot

Using ggplot to visualize our findings

Hey-ho! That looks pretty damn good. Now let's plot this and call it a day.

```
ggplot(data = popular_cats_and_dogs,
       aes(x = reorder(animals_name, Total), y = Total)) +
  geom_bar(stat = "identity",
           fill = "#b39ddb",
           width = .9,
           position = position_dodge(width=0.75)) +
  labs(title = 'Is that a dog or a cat?',
       caption = '#TidyTuesday by Dash Wieland',
       subtitle = "Seattle's most popular pet names given to dogs and cats in equal numbers") +
  ylab('Number of Pets') +
  xlab("Pet Name") +
  theme(plot.background = element_rect(fill = "#FDF5FA"),
        panel.background = element_rect(fill = "#FDF5FA",
                                        colour = "#FDF5FA",
                                        size = 0.5, linetype = "solid"),
        panel.grid.major = element_line(size = 0.15, linetype = 'solid',
                                        colour = "#DCAED1"),
```

```
      panel.grid.minor = element_line(size = 0.05, linetype = 'solid',
                                      colour = "#DCAED1"),
      plot.title = element_text(color="#4A2040", size=16, face="bold"),
      plot.subtitle = element_text(color="#4A2040", size=12),
      axis.title.x = element_text(color="#4A2040", size=10),
      axis.title.y = element_text(color="#4A2040", size=10)
) +
coord_flip()
```