# Predicting pet adoption speeds

*Dennis Sim*

*31 March, 2019*

## Contents

# 1 Introduction

PetFinder.my is an animal welfare platform in Malaysia with a database of more than 150,000 animals where animals are listed for adoption. This project will study the possibility to build a prediction model that predicts animal adoption rates which is usually correlated to the metadata associated with the pets' listing profiles.

## 1.1 Data set

The data set used for this project is made up of 4 csv files that are available through Kaggle. This is a smaller subset of files that are originally used for a Kaggle competition.

These files contain profiles of pets listed for adoption along with metadata such as age, species, breed, gender, color, fur length, maturity size, health information (including sterilization, deworming and vaccination status), location, descriptive text and photo of the pet.

## 1.2 Project goal summary

This study will firstly attempt to explore the data set and seek to better understand the relationships between the metadata that is associated with the pets' listing profile and the pets' adoption speeds.

The goal of this project is to predict the adoptability of pets - specifically, how quickly would a pet adopted based on the metadata associated with their online profiles.

## 1.3 Key steps performed

The data set is firstly explored to ensure it is clean and in a suitable state for experimenting with different methods to build the prediction model. The data set is also visualized in different ways to better understand the characteristics of potential features that could be used as predictors in building the prediction model.

Random forest method is chosen for building this prediction model as it is close to human decision-making process with parameters such as mtry (number of variables randomly sampled as candidates at each split) and ntree (number of trees) that can be tuned to refine the model. Another reason for choosing random forest is also due to the reason that the predictors of this data set is made up of categorical and numerical types, where some other machine learning algorithms may not be suitable under such circumstances.

# 2 Methods and Analysis

This section describes in detail the process and techniques used for data cleaning, data exploration and visualization, insights gained, and discusses modelling approaches.

## 2.1 Data source

There are 4 csv files that make up the data set for this project as described in the previous section

| Filename | Description | Count |
|---|---|---|
| pets.csv | Contains Adoption Speed and pet characteristics | 14993 |
| breed_labels.csv | Contains Type, and BreedName for each BreedID. Type 1 = dog, 2 = cat | 307 |
| color_labels.csv | Contains ColorName for each ColorID | 7 |
| state_labels.csv | Contains StateName for each StateID | 15 |

## 2.2 Data fields

The pets.csv data set contains 14993 samples with 24 columns including the dependent column *AdoptionSpeed* that is to be predicted.

```
str(df_pets)
```

```
## 'data.frame': 14993 obs. of 24 variables:
## $ Type : int 2 2 1 1 1 2 2 1 2 2 ...
## $ Name : Factor w/ 9061 levels "","!","! Med Long Fur Kittens",..: 5989 6044
1505 5420 3698 1 910 7557 1 4405 ...
## $ Age : int 3 1 1 4 1 3 12 0 2 12 ...
## $ Breed1 : int 299 265 307 307 307 266 264 307 265 265 ...
## $ Breed2 : int 0 0 0 0 0 0 264 0 0 0 ...
## $ Gender : int 1 1 1 2 1 2 1 2 2 2 ...
## $ Color1 : int 1 1 2 1 1 5 1 1 6 1 ...
## $ Color2 : int 7 2 7 2 0 6 0 2 0 7 ...
## $ Color3 : int 0 0 0 0 0 0 0 7 0 0 ...
## $ MaturitySize : int 1 2 2 2 2 2 2 2 2 2 ...
## $ FurLength : int 1 2 2 1 1 1 3 1 2 2 ...
## $ Vaccinated : int 2 3 1 1 2 2 2 2 2 3 ...
## $ Dewormed : int 2 3 1 1 2 2 2 2 2 3 ...
## $ Sterilized : int 2 3 2 2 2 2 3 2 2 3 ...
## $ Health : int 1 1 1 1 1 1 1 1 1 1 ...
## $ Quantity : int 1 1 1 1 1 1 1 6 1 1 ...
## $ Fee : int 100 0 0 150 0 0 300 0 0 0 ...
## $ State : int 41326 41401 41326 41401 41326 41326 41326 41326 41326 41326
...
## $ RescuerID : Factor w/ 5595 levels "0007e457eb0583479bb888d54764911f",..:
2933 1114 5486 3233 3298 817 683 724 4747 711 ...
## $ VideoAmt : int 0 0 0 0 0 0 0 0 0 0 ...
## $ Description : Factor w/ 14033 levels " McD. Loving & docile. Gives kisses.
Manja. She will make you fall in love with her sweet nature. Vaccinated &
spayed.",..: 7940 5544 10558 3889 11164 11250 12839 9805 13049 12169 ...
## $ PetID : Factor w/ 14993 levels "0008c5398","000a290e4",..: 7758 5717 3087
5149 7642 12297 1709 8793 11271 7041 ...
## $ PhotoAmt : num 1 2 7 8 3 2 3 9 6 2 ...
```

```
## $ AdoptionSpeed: int 2 0 3 2 2 2 1 3 1 4 ...
```

The data fields in from the pets.csv file is described below:

- PetID - Unique hash ID of pet profile
- AdoptionSpeed - Categorical speed of adoption. Lower is faster. This is the value to predict. See below section for more info.
- Type - Type of animal (1 = Dog, 2 = Cat)
- Name - Name of pet (Empty if not named)
- Age - Age of pet when listed, in months
- Breed1 - Primary breed of pet (Refer to *df_breed*)
- Breed2 - Secondary breed of pet, if pet is of mixed breed (Refer to *df_breed*)
- Gender - Gender of pet (1 = Male, 2 = Female, 3 = Mixed, if profile represents group of pets)
- Color1 - Color 1 of pet (Refer to *df_color*)
- Color2 - Color 2 of pet (Refer to *df_color*)
- Color3 - Color 3 of pet (Refer to *df_color*)
- MaturitySize - Size at maturity (1 = Small, 2 = Medium, 3 = Large, 4 = Extra Large, 0 = Not Specified)
- FurLength - Fur length (1 = Short, 2 = Medium, 3 = Long, 0 = Not Specified)
- Vaccinated - Pet has been vaccinated (1 = Yes, 2 = No, 3 = Not Sure)
- Dewormed - Pet has been dewormed (1 = Yes, 2 = No, 3 = Not Sure)
- Sterilized - Pet has been spayed / neutered (1 = Yes, 2 = No, 3 = Not Sure)
- Health - Health Condition (1 = Healthy, 2 = Minor Injury, 3 = Serious Injury, 0 = Not Specified)
- Quantity - Number of pets represented in profile
- Fee - Adoption fee (0 = Free)
- State - State location in Malaysia (Refer to *df_state*)
- RescuerID - Unique hash ID of rescuer
- VideoAmt - Total uploaded videos for this pet
- PhotoAmt - Total uploaded photos for this pet
- Description - Profile write-up for this pet. The primary language used is English, with some in Malay or Chinese.

The classes for pet adoption speed is originally set as 0, 1, 2, 3 and 4. This is associated in a relationship whereby the larger the number, the slower it takes for a pet to get adopted.

To facilitate data visualization for this project, the numbers of the classes are reversed in such a way that the larger the number, the faster the pet is adopted.

- 0: No adoption after 100 days
- 1: 31-90 Days
- 2: 8-30 Days
- 3: 1-7 Days
- 4: Same Day

### 2.3 Data cleaning

There are 307 breed types for all dogs and cats including “Mixed Breed”. Type=1 indicates a dog breed and Type=2 indicates a cat breed. In the pet profile *df_pets* data frame, the BreedID is used

in column Breed1 (primary breed) and Breed2 (secondary breed).

```
##     BreedID Type      BreedName
## 302     302    2         Torbie
## 303     303    2  Tortoiseshell
## 304     304    2 Turkish Angora
## 305     305    2    Turkish Van
## 306     306    2         Tuxedo
## 307     307    1    Mixed Breed
```

```
##        PetID Type Breed1 Breed2
## 1  86e1089a3    2    299      0
## 2  6296e909a    2    265      0
## 3  3422e4906    1    307      0
## 4  5842f1ff5    1    307      0
## 5  850a43f90    1    307      0
## 6  d24c30b4b    2    266      0
## 7  1caa6fcdb    2    264    264
## 8  97aa9eeac    1    307      0
## 9  c06d167ca    2    265      0
## 10 7a0942d61    2    265      0
```

It is possible for a pet to be a mix of different breeds and very often the pet profile will either contain different BreedIDs for Breed1 and Breed2; or be listed with BreedID 307 on either one of Breed1 or Breed2. Mixed Breed is denoted by BreedID 307 and this ID be used in either column Breed1 or Breed2, or sometimes both.

```
##   BreedID Type   BreedName
## 1     307    1 Mixed Breed
```

```
##       PetID Breed1 Breed2
## 1 3422e4906    307      0
## 2 5842f1ff5    307      0
## 3 850a43f90    307      0
## 4 97aa9eeac    307      0
## 5 8b693ca84    307      0
## 6 aaedd873d    307      0
```

```
##       PetID Breed1 Breed2
## 1 f9d07d5fa    307    307
## 2 671ffc570    128    307
## 3 9609a0b86    307    307
## 4 b0dec8779    213    307
## 5 aaf4a1439    307    307
## 6 35593da00    307    307
```

A pure breed pet will have Breed1 value associated with a particular BreedID and Breed2 contains 0. This indicates the pet is a pure breed. However, if Breed1 has a value of 307, it indicates the pet is a mixed breed and this will be dealt with in the later section.

There are also several pet profiles with Breed2 associated with a BreedID and whilst having Breed1 containing 0. This will also be fixed later for consistency.

```
df_pets %>%
  filter(Breed1 != 0 & Breed2 == 0) %>%
  select(PetID, Breed1, Breed2) %>% head()
```

```
##       PetID Breed1 Breed2
## 1 86e1089a3    299      0
## 2 6296e909a    265      0
## 3 3422e4906    307      0
## 4 5842f1ff5    307      0
## 5 850a43f90    307      0
## 6 d24c30b4b    266      0
```

```
df_pets %>%
  filter(Breed1 == 0 & Breed2 != 0) %>%
  select(PetID, Breed1, Breed2) %>% head()
```

```
##       PetID Breed1 Breed2
## 1 375905770      0     26
## 2 da8d4a273      0    307
## 3 27e74e45c      0    266
## 4 7b5bee232      0    307
## 5 0327b8e94      0    205
```

However, it is good to see that there is no pet profile containing both Breed1 and Breed2 having 0 values. So the data set is clean on this aspect.

```
df_pets %>% filter(Breed1 == 0 & Breed2 == 0) %>% select(PetID, Breed1, Breed2)
```

```
## [1] PetID  Breed1 Breed2
## <0 rows> (or 0-length row.names)
```

As mentioned above, there are several rows with Breed1 having 0 values and Breed2 having a non-zero value denoting a particular breed type. The values for Breed1 and Breed2 are interchanged for consistency to have Breed1 to indicate the primary breed, and Breed2 to have value 0 if the pet is a pure breed.

```
## [1] PetID  Breed1 Breed2
## <0 rows> (or 0-length row.names)
```

There are also pets listed with both Breed1 (primary breed) and Breed2 (secondary breed) having the same value. In such case, Breed2 will be set to zero. A new column Breed_Mix is also introduced to the *df_pets* data frame to indicate if the pet is a mixed breed or pure breed.

```
##        PetID Breed1 Breed2
## 1  1caa6fcdb    264    264
## 2  f9d07d5fa    307    307
## 3  9609a0b86    307    307
## 4  aaf4a1439    307    307
## 5  35593da00    307    307
## 6  22051ab84    266    266
## 7  573b3ec3d    307    307
```

```
## 8  a6ac8884b    307    307
## 9  19b1f4263    266    266
## 10 80ebea85c    307    307
```

```
## [1] PetID  Breed1 Breed2
## <0 rows> (or 0-length row.names)
```

In addition, another 3 new columns are added to the df_pets data frame to facilitate data visualization in the next section.

- Breed1_Name - label for Breed1
- Breed2_Name - label for Breed2
- Breed_Combined - combined labels for Breed1 and Breed2

```
## Source: local data frame [10 x 5]
## Groups: <by row>
##
## # A tibble: 10 x 5
##    Breed1 Breed2 Breed1_Name           Breed2_Name Breed_Combined
##     <int>  <dbl> <chr>                 <chr>       <chr>
##  1    299      0 Sphynx (hairless cat) <NA>        Sphynx (hairless cat)
##  2    265      0 Domestic Long Hair    <NA>        Domestic Long Hair
##  3    307      0 Tuxedo                <NA>        Tuxedo
##  4    307      0 Tuxedo                <NA>        Tuxedo
##  5    307      0 Tuxedo                <NA>        Tuxedo
##  6    266      0 Domestic Medium Hair  <NA>        Domestic Medium Hair
##  7    264      0 Dilute Tortoiseshell  <NA>        Dilute Tortoiseshell
##  8    307      0 Tuxedo                <NA>        Tuxedo
##  9    265      0 Domestic Long Hair    <NA>        Domestic Long Hair
## 10    265      0 Domestic Long Hair    <NA>        Domestic Long Hair
```

There is a Description column that contains text description of the listed pet. The effects of text semantics is outside the scope of this project. However, another new column Desc_WordCount is added to *df_pets* data frame that store the number of words in the description to facilitate data visualization in the following sections.

```
# add word count of Description
df_pets <- df_pets %>%
  mutate(Description = as.character(Description),
         Desc_WordCount = sapply(strsplit(Description, " "), length))
```

```
# df_pets data structure
class(df_pets$Desc_WordCount)
```

```
## [1] "integer"
```

```
str(df_pets$Desc_WordCount)
```

```
##  int [1:14993] 69 23 69 25 81 18 78 20 13 45 ...
```

## 2.4 Data exploration, visualization and insights gained

In this section, the data set is explored in detail and visualized to better understand the data set and the characteristics of metadata that is associated with the pet profiles.

```
head(df_breed)
```

```
##   BreedID Type        BreedName
## 1       1    1    Affenpinscher
## 2       2    1     Afghan Hound
## 3       3    1 Airedale Terrier
## 4       4    1           Akbash
## 5       5    1            Akita
## 6       6    1 Alaskan Malamute
```

```
head(df_color)
```

```
##   ColorID ColorName
## 1       1     Black
## 2       2     Brown
## 3       3    Golden
## 4       4    Yellow
## 5       5     Cream
## 6       6      Gray
```

```
head(df_state)
```

```
##   StateID    StateName
## 1   41336        Johor
## 2   41325        Kedah
## 3   41367     Kelantan
## 4   41401 Kuala Lumpur
## 5   41415       Labuan
## 6   41324       Melaka
```

```
as_tibble(head(df_pets[1:10]))
```

```
## # A tibble: 6 x 10
##    Type Name     Age Breed1 Breed2 Gender Color1 Color2 Color3 MaturitySize
##   <int> <fct>  <int>  <int>  <dbl>  <int>  <int>  <int>  <int>        <int>
## 1     2 Nibble     3    299      0      1      1      7      0            1
## 2     2 No Na~     1    265      0      1      1      2      0            2
## 3     1 Brisco     1    307      0      1      2      7      0            2
## 4     1 Miko       4    307      0      2      1      2      0            2
## 5     1 Hunter     1    307      0      1      1      0      0            2
## 6     2 ""         3    266      0      2      5      6      0            2
```

```
as_tibble(head(df_pets[11:18]))
```

```
## # A tibble: 6 x 8
##   FurLength Vaccinated Dewormed Sterilized Health Quantity   Fee State
```

```
##       <int>      <int>    <int>      <int>  <int>    <int> <int> <int>
## 1         1          2        2          2      1        1   100 41326
## 2         2          3        3          3      1        1     0 41401
## 3         2          1        1          2      1        1     0 41326
## 4         1          1        1          2      1        1   150 41401
## 5         1          2        2          2      1        1     0 41326
## 6         1          2        2          2      1        1     0 41326
```

```
head(select(df_pets, 19,20, 22, 23,29))
```

```
## Source: local data frame [6 x 5]
## Groups: <by row>
##
## # A tibble: 6 x 5
##   RescuerID                        VideoAmt PetID    PhotoAmt Desc_WordCount
##   <fct>                               <int> <fct>       <dbl>          <int>
## 1 8480853f516546f6cf33aa88cd76c3~         0 86e1089~        1             69
## 2 3082c7125d8fb66f7dd4bff4192c8b~         0 6296e90~        2             23
## 3 fa90fa5b1ee11c86938398b60abc32~         0 3422e49~        7             69
## 4 9238e4f44c71a75282e62f7136c6b2~         0 5842f1f~        8             25
## 5 95481e953f8aed9ec3d16fc4509537~         0 850a43f~        3             81
## 6 22fe332bf9c924d4718005891c63fb~         0 d24c30b~        2             18
```

```
as_tibble(head(df_pets[24:28]))
```

```
## # A tibble: 6 x 5
##   AdoptionSpeed Breed_Mix Breed1_Name        Breed2_Name Breed_Combined
##           <int> <chr>     <chr>              <chr>       <chr>
## 1             2 Pure      Sphynx (hairless ~ <NA>        Sphynx (hairless ~
## 2             4 Pure      Domestic Long Hair <NA>        Domestic Long Hair
## 3             1 Mixed     Tuxedo             <NA>        Tuxedo
## 4             2 Mixed     Tuxedo             <NA>        Tuxedo
## 5             2 Mixed     Tuxedo             <NA>        Tuxedo
## 6             2 Pure      Domestic Medium H~ <NA>        Domestic Medium H~
```

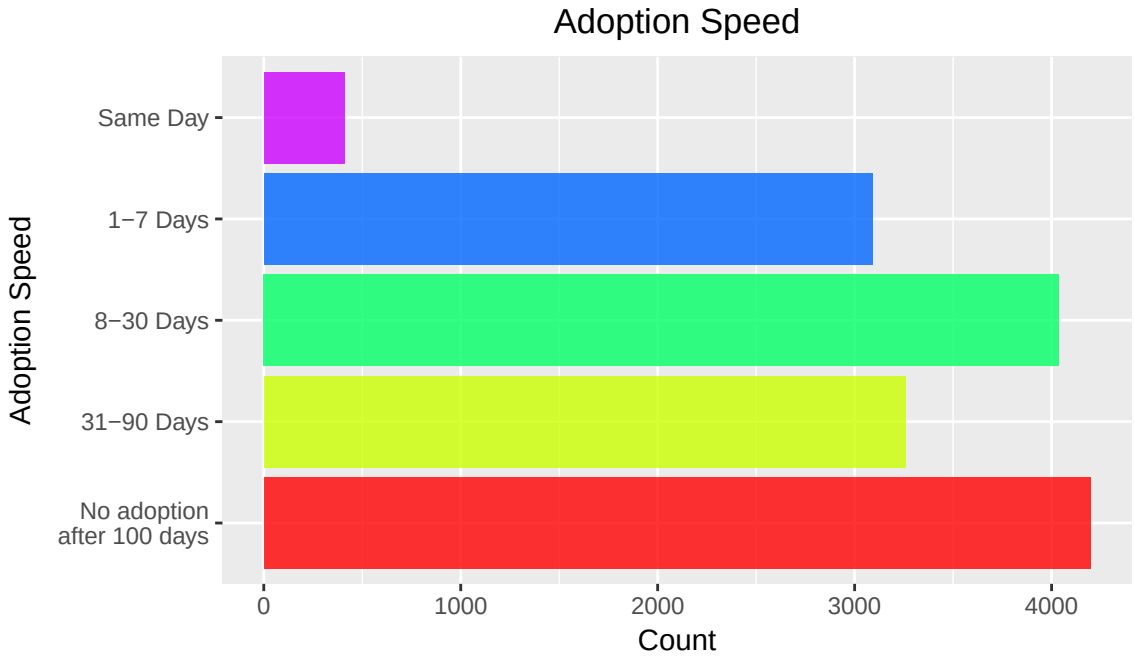
```
as_tibble(head(df_pets[21]))
```

```
## # A tibble: 6 x 1
##   Description
##   <chr>
## 1 Nibble is a 3+ month old ball of cuteness. He is energetic and playful. ~
## 2 I just found it alone yesterday near my apartment. It was shaking so I h~
## 3 Their pregnant mother was dumped by her irresponsible owner at the roads~
## 4 Good guard dog, very alert, active, obedience waiting for her good maste~
## 5 This handsome yet cute boy is up for adoption. He is the most playful pa~
## 6 This is a stray kitten that came to my house. Have been feeding it, but ~
```

### 2.4.1 Adoption speed

A small fraction of pets get adopted on the same day it was listed. This could be due to chance (a lucky pet or a lucky adopter) or people just tend to like "newly" listed pets sometimes. The other 4 classes where pets get adopted between 1 to 90 days, and sometimes not adopted at all after 100 days have a somewhat closer distribution with each other.

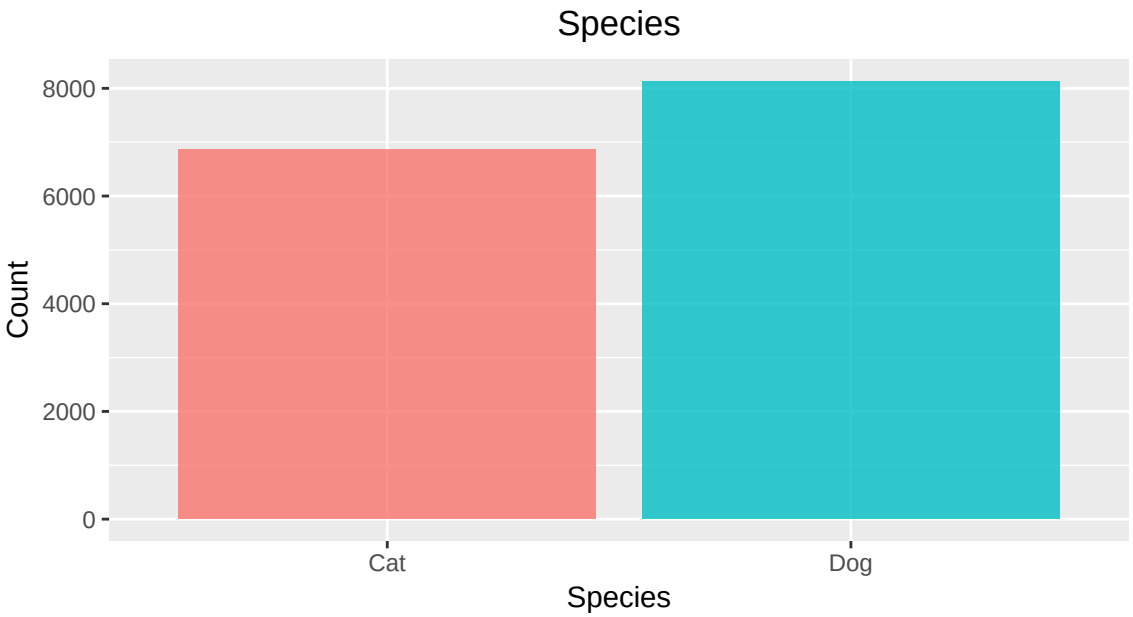
```
## # A tibble: 5 x 3
##   AdoptionSpeed Count   Rate
##   <chr>         <int>  <dbl>
## 1 4               410 0.0273
## 2 3              3090 0.206
## 3 2              4037 0.269
## 4 1              3259 0.217
## 5 0              4197 0.280
```



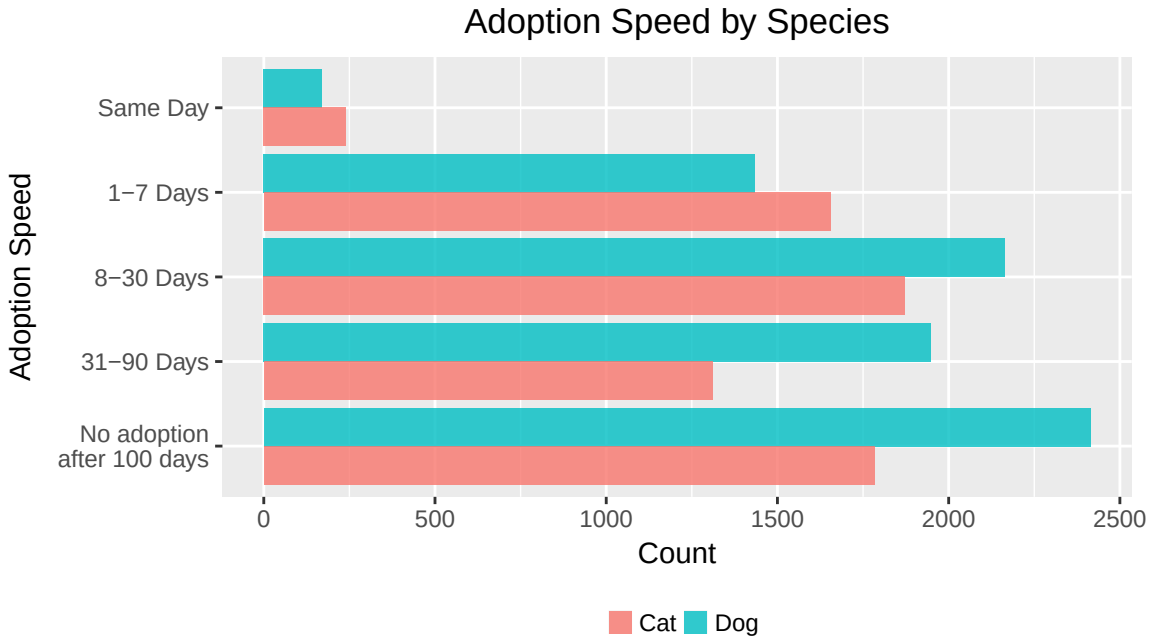

### 2.4.2 Species / Type

Cats appear to have a higher adoption chance than dogs within the first week of listing. There are more dogs than cats that are not adopted after 100 days.

```
## # A tibble: 2 x 2
##   Species Count
##   <chr>   <int>
## 1 Dog      8132
## 2 Cat      6861
```




*There are 18.5% more dogs than cats*




```
## # A tibble: 2 x 3
##   Species AvgSpeed Count
##   <chr>      <dbl> <int>
## 1 Cat         1.60  6861
## 2 Dog         1.38  8132
```

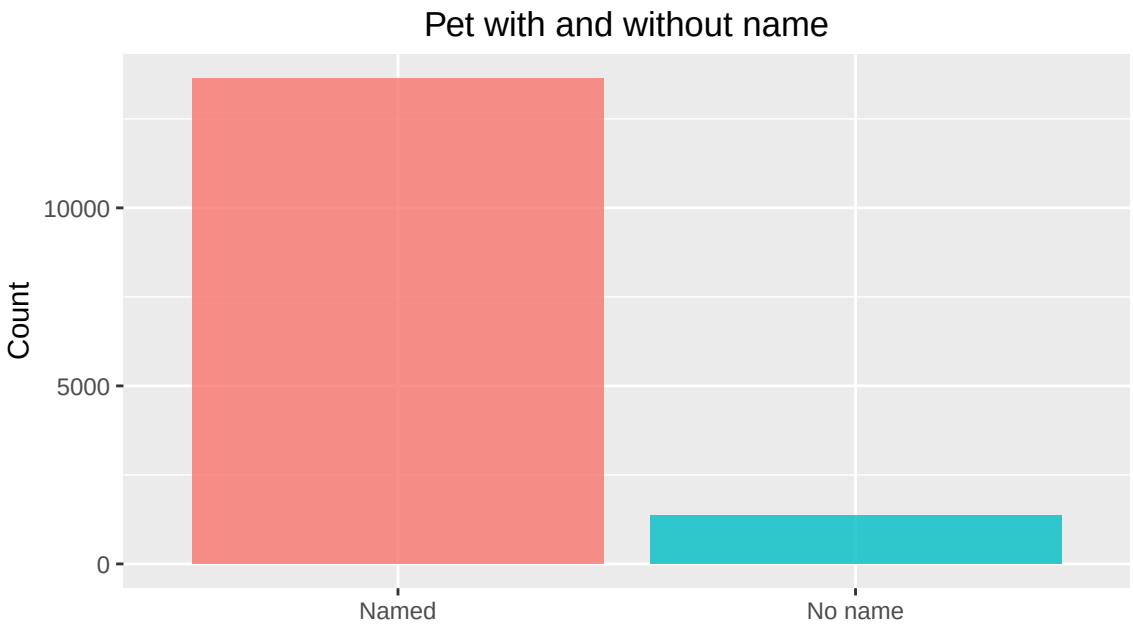
However, there are less cats are available for adoption compared to dogs. There are more dogs (+18.5%) than cats. Cats also have a faster average adoption speed, perhaps people prefer cats to

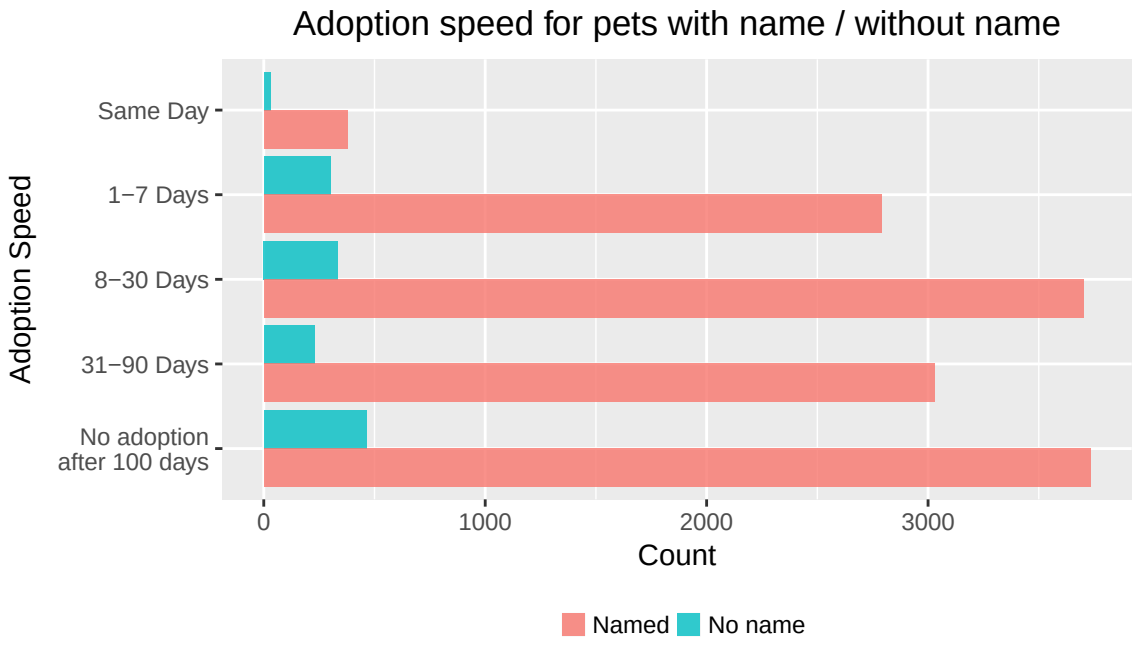
dogs. We shall see in the following sections.

### 2.4.3 Named and unnamed pets

Named pets appear to have a higher chance of being adopted on the same day it is listed.

```
## Source: local data frame [1,362 x 1]
## Groups: <by row>
##
## # A tibble: 1,362 x 1
##    Name
##    <fct>
##  1 No Name Yet
##  2 ""
##  3 ""
##  4 ""
##  5 No Name
##  6 No Name
##  7 ""
##  8 ""
##  9 ""
## 10 ""
## # ... with 1,352 more rows
```





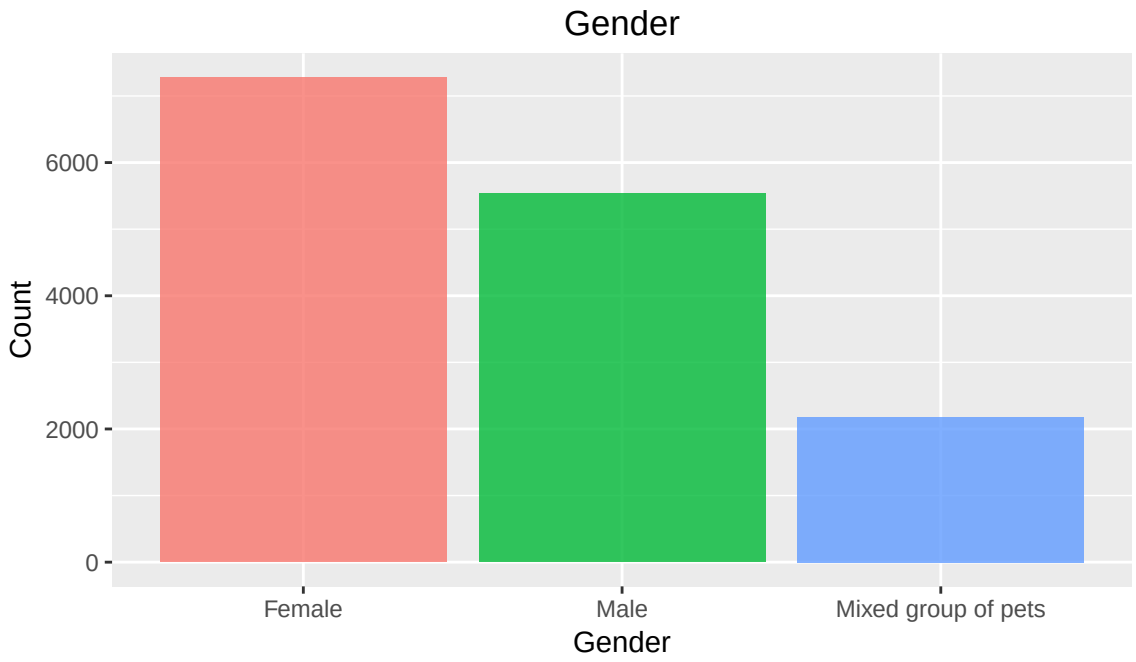

```
head(avg_speed_name)
```

```
## # A tibble: 2 x 3
##   HasName AvgSpeed Count
##   <chr>      <dbl> <int>
## 1 Named       1.49 13631
## 2 No name     1.42  1362
```

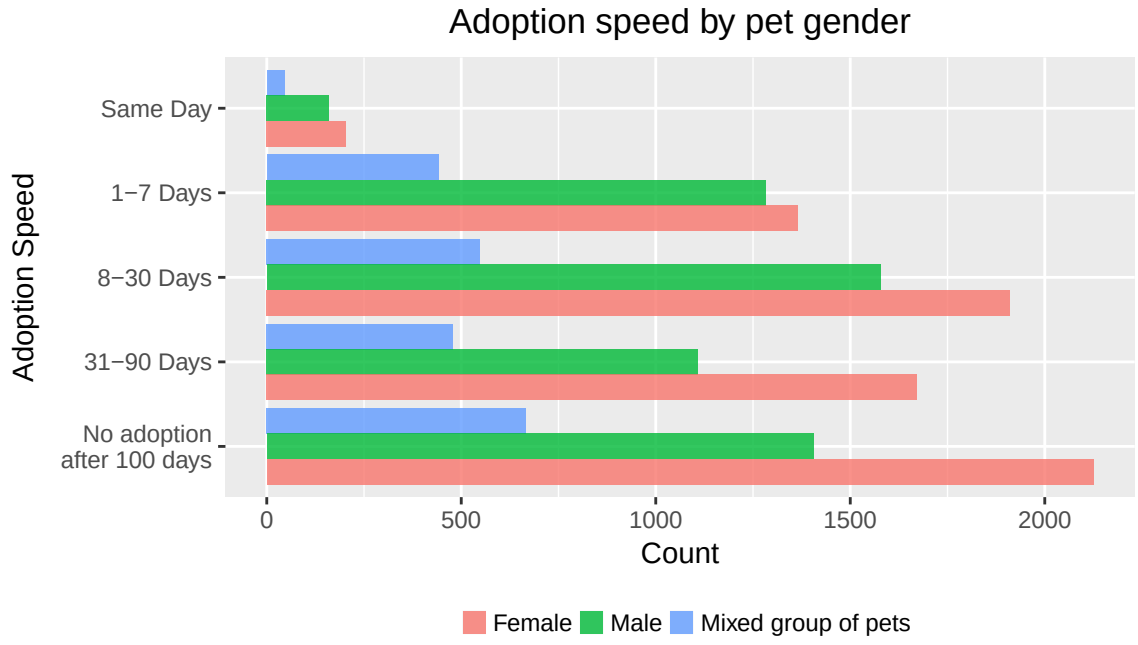
After that, the name does not appear to have effect on adoption speed of the pet. However, only 9% of pets are listed without names. On average, the adoption speed of pets listed with and without name is about the same. Pets listed with or without name do not have much effect on adoption speed.

### 2.4.4 Gender

Gender can be male, female or mixed (if the pet profile consists of a group of pets of different genders).

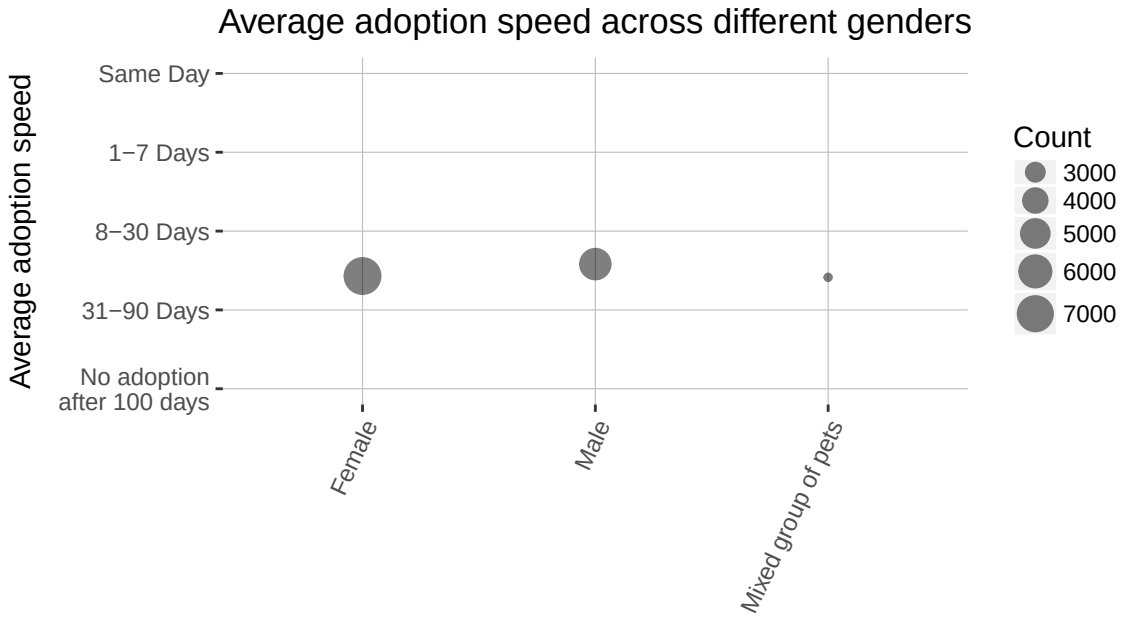



```
## # A tibble: 3 x 3
##   Gender Count PetGender
##    <int> <int> <chr>
## 1      1  5536 Male
## 2      2  7277 Female
## 3      3  2180 Mixed group of pets
```




```
## # A tibble: 3 x 3
```

```
##   Gender              AvgSpeed Count
##   <chr>                  <dbl> <int>
## 1 Male                    1.58  5536
## 2 Female                  1.43  7277
## 3 Mixed group of pets     1.41  2180
```



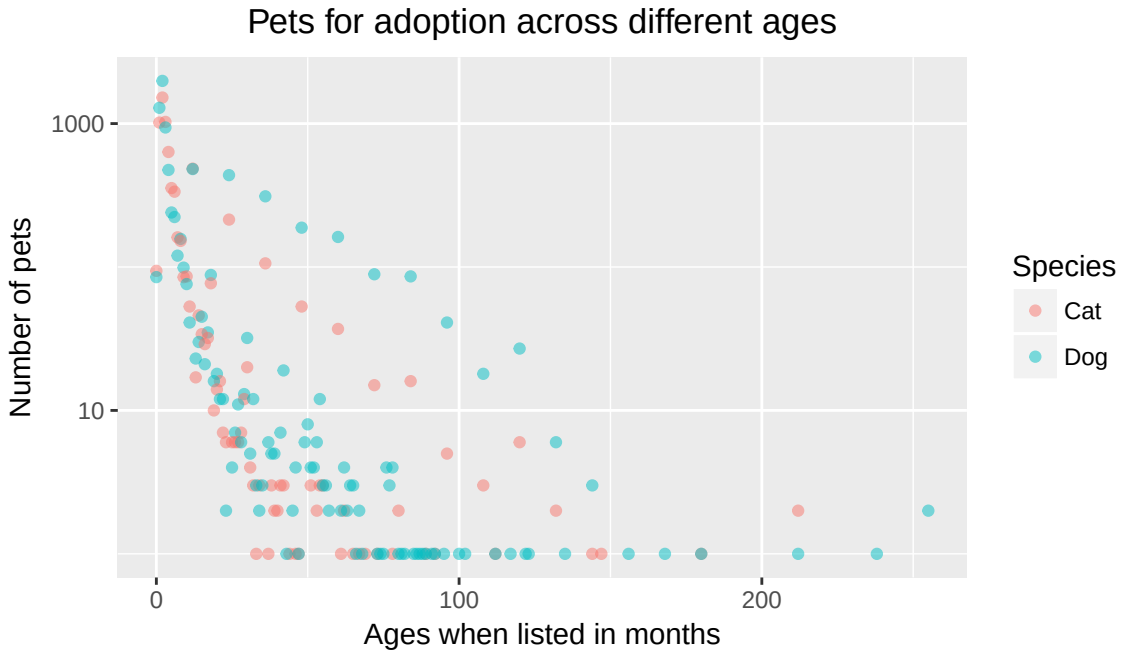

Multiple pets listed as a group entry makes up 15% of the data set. Female pets appear to be more popular of being adopted. However, there are more female pets in the population. The average adoption speed by gender for each adoption_speed class shows all three classes of gender having similar average adoption speed.

### 2.4.5 Age

Most pets are listed for adoption when young, mostly between 0-4 months. There is also a good portion of pets older than 12 months that are listed in multiples of 12 (denoting 12, 24, 36 months) which suggests the person who list the pets for adoption do not bother to describe the pet's age to the month, and rather make a direct year to month conversion.

```
## # A tibble: 20 x 3
## # Groups:   Species [2]
##    Species   Age Count
##    <chr>   <int> <int>
##  1 Dog         2  1984
##  2 Cat         2  1519
##  3 Dog         1  1288
##  4 Cat         3  1027
##  5 Cat         1  1016
##  6 Dog         3   939
##  7 Cat         4   634
```

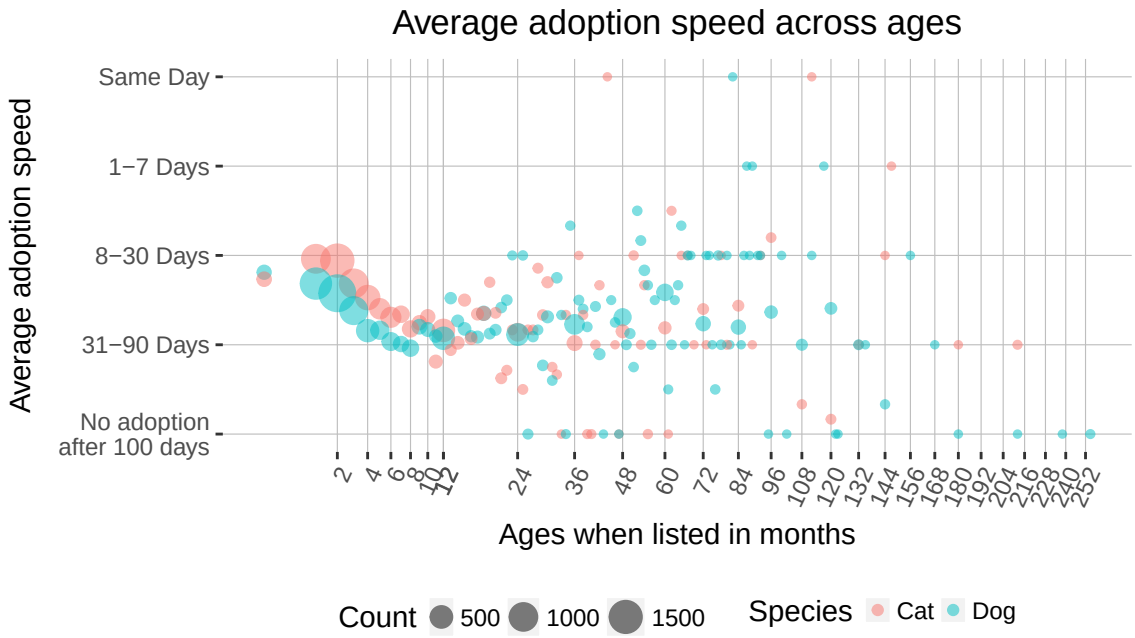
```
##  8 Cat         12   485
##  9 Dog         12   482
## 10 Dog          4   475
## 11 Dog         24   437
## 12 Cat          5   355
## 13 Cat          6   335
## 14 Dog         36   311
## 15 Dog          5   240
## 16 Dog          6   223
## 17 Cat         24   214
## 18 Dog         48   188
## 19 Dog         60   162
## 20 Cat          7   161
```




```
## Selecting by Count
```

```
## # A tibble: 40 x 4
## # Groups:   Species [2]
##    Species   Age AvgSpeed Count
##    <chr>   <int>    <dbl> <int>
##  1 Cat         1     1.96  1016
##  2 Cat         2     1.94  1519
##  3 Dog         0     1.81    85
##  4 Cat         0     1.73    94
##  5 Cat         3     1.69  1027
##  6 Dog         1     1.68  1288
##  7 Dog        60     1.59   162
##  8 Dog         2     1.58  1984
```

```
##  9 Cat          4     1.53   634
## 10 Cat         15     1.5     34
## # ... with 30 more rows
```



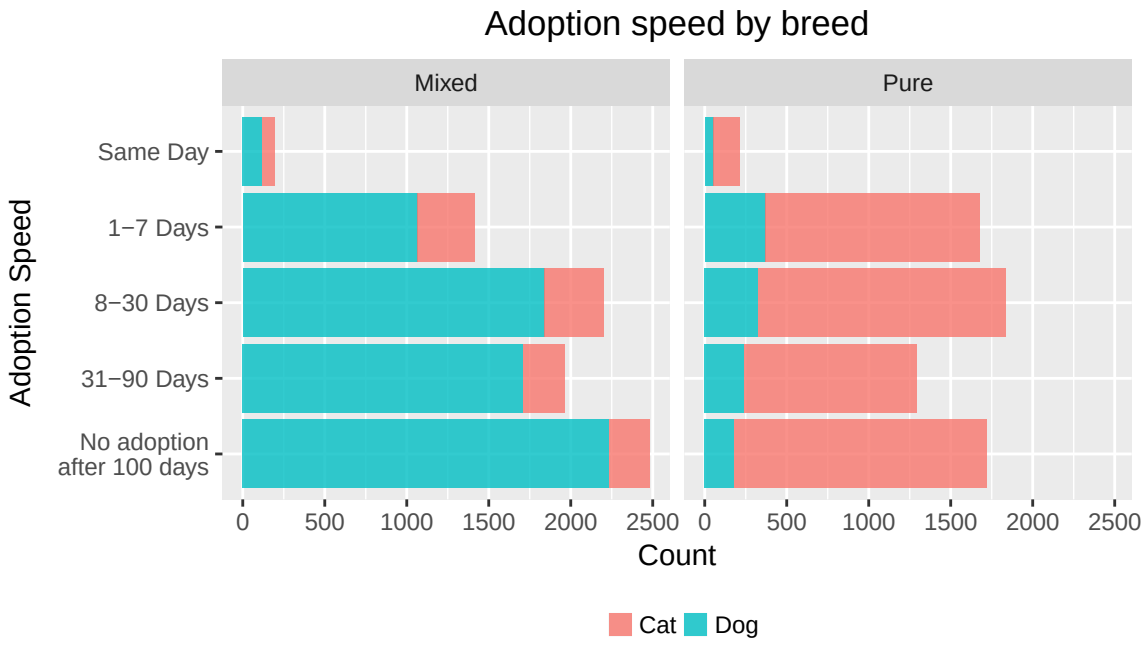

Many young pets below 12 months are quickly adopted and mostly adopted. Some pets older than 24 months end up not adopted after 100 days.

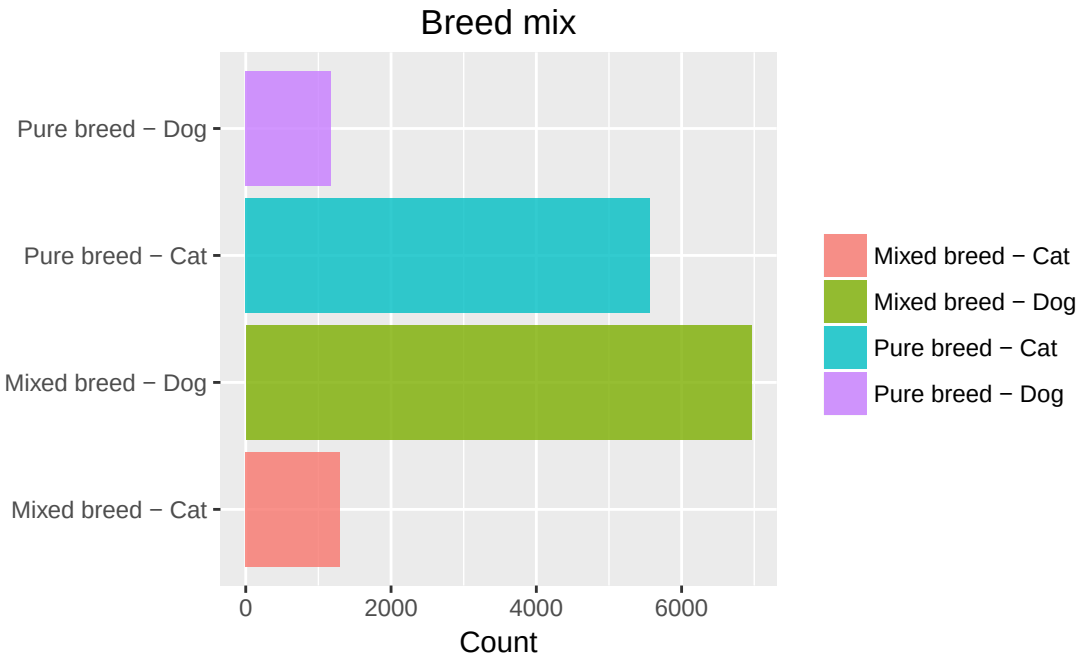
It is interesting to see that there are some (2%) pets older than 24 moths are adopted within 1 week. This is a small percentage (2%) and can be ignored.

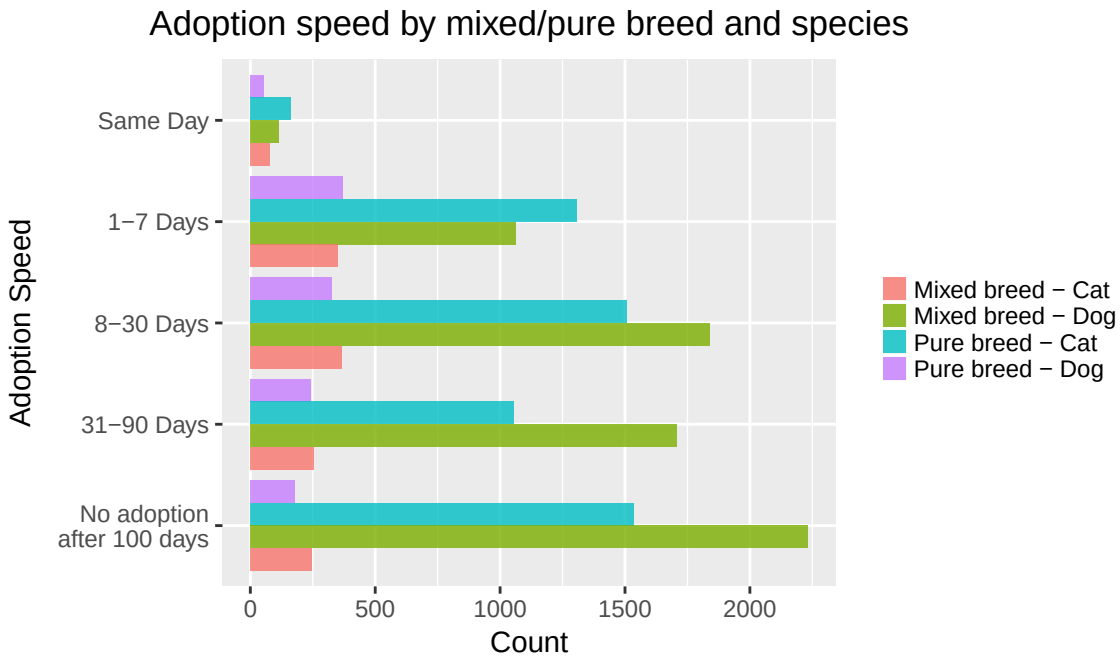
### 2.4.6 Breed

```
## # A tibble: 4 x 3
## # Groups:   Breed_Mix [2]
##   Breed_Mix Species Count
##   <chr>     <chr>   <int>
## 1 Mixed     Cat      1295
## 2 Mixed     Dog      6961
## 3 Pure      Cat      5566
## 4 Pure      Dog      1171
```

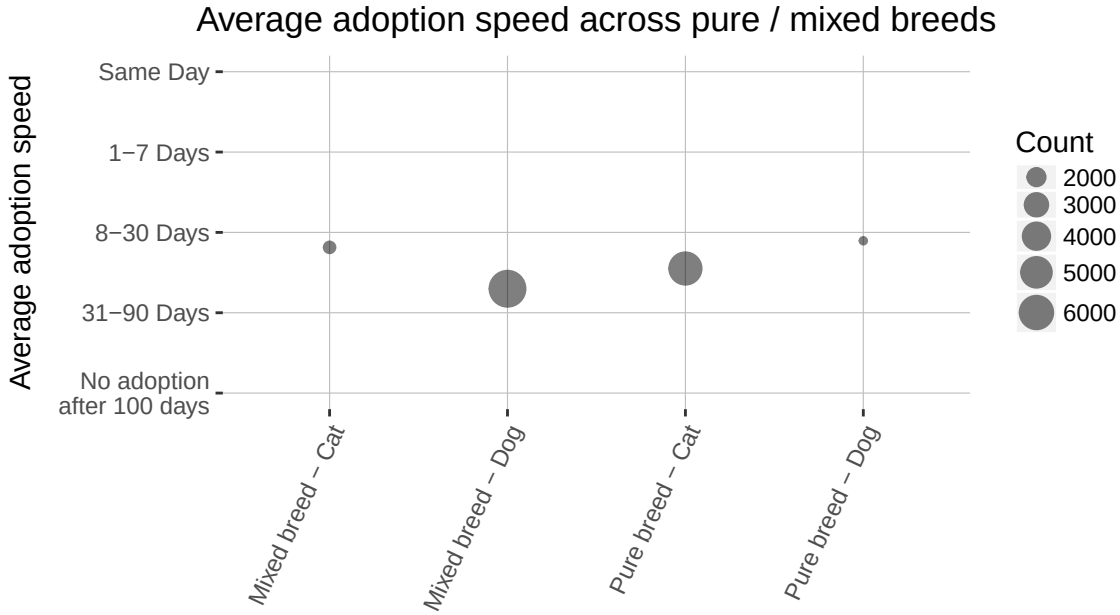
There are 45% of pets that are pure breeds - with 81% being pure breed cats and 81% being pure breed dogs.


Adoption speed by breed
Mixed
Pure
Same Day
1−7 Days
8−30 Days
31−90 Days
No adoption after 100 days
Adoption Speed
0
500
1000
1500
2000
2500
Count
Cat
Dog


Breed mix
Pure breed − Dog
Pure breed − Cat
Mixed breed − Dog
Mixed breed − Cat
0
2000
4000
6000
Count
Mixed breed − Cat
Mixed breed − Dog
Pure breed − Cat
Pure breed − Dog

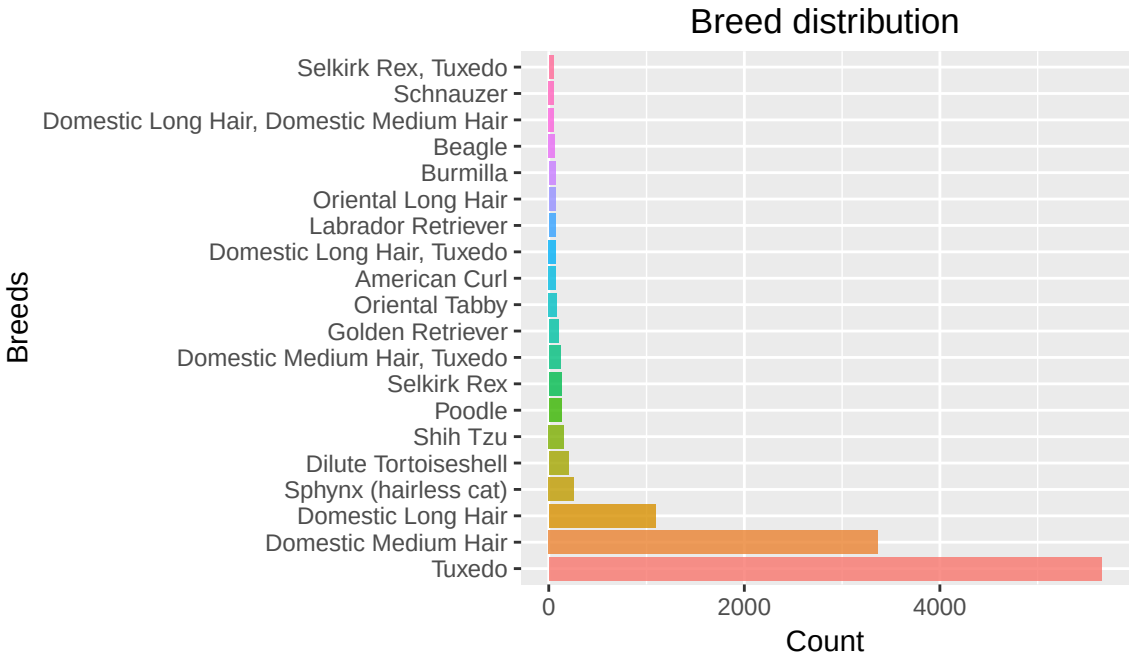



```
## # A tibble: 4 x 3
##   Breed_Species     AvgSpeed Count
##   <chr>                <dbl> <int>
## 1 Pure breed - Dog      1.90  1171
## 2 Mixed breed - Cat     1.81  1295
## 3 Pure breed - Cat      1.55  5566
## 4 Mixed breed - Dog     1.30  6961
```



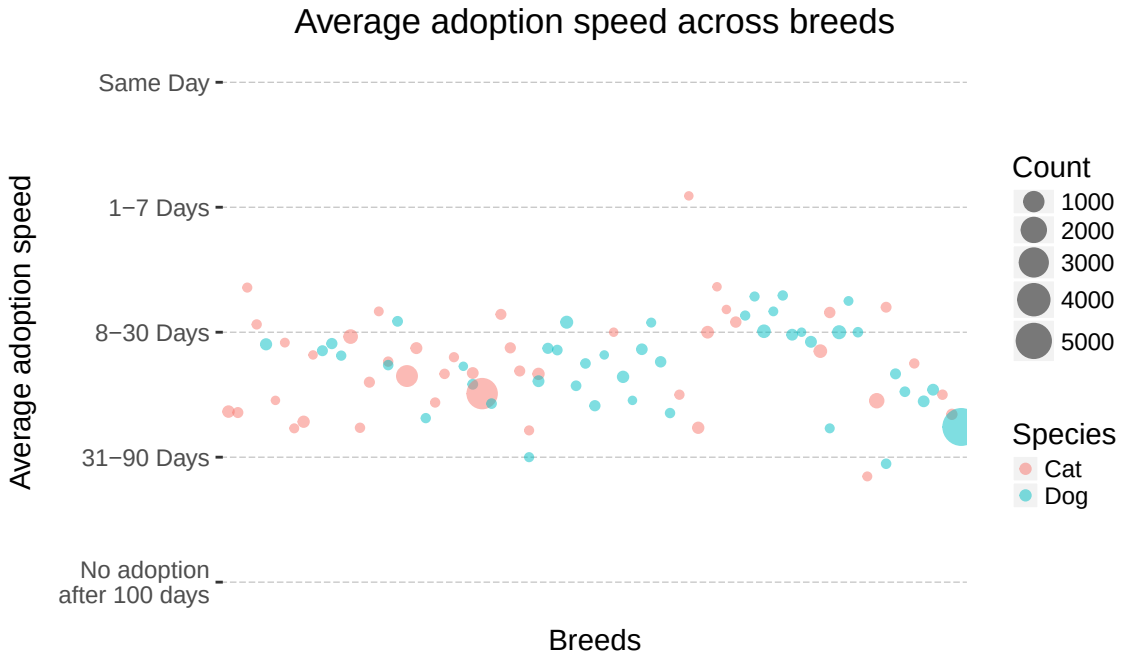
Pure breed dogs appear to have faster average adoption speed than pure breed cats, followed by mixed breed dogs.

It is also interesting to note that most cats are pure breed, and there's a small number of mixed breed cats not adopted for $> 100$ days.




```
## # A tibble: 20 x 4
## # Groups:   Breed_Combined [20]
##    Breed_Combined                       Species AvgSpeed Count
##    <chr>                                <chr>      <dbl> <int>
##  1 Abyssinian, Terrier                  Cat            4     1
##  2 Akita, Sphynx (hairless cat)         Cat            4     1
##  3 American Curl, Munchkin              Cat            4     1
##  4 Balinese, Golden Retriever           Cat            4     1
##  5 Balinese, Ocicat                     Cat            4     1
##  6 Balinese, Sheep Dog                  Cat            4     1
##  7 Beagle, LaPerm                       Cat            4     1
##  8 British Shorthair, Poodle            Dog            4     1
##  9 Bull Terrier, Domestic Medium Hair   Cat            4     1
## 10 Burmilla, Dilute Tortoiseshell       Cat            4     1
## 11 Burmilla, Rottweiler                 Cat            4     1
## 12 Chinese Crested Dog, Tuxedo          Dog            4     1
## 13 Collie, Tuxedo                       Cat            4     1
## 14 Dachshund, Oriental Tabby            Cat            4     1
## 15 Domestic Medium Hair, Poodle         Dog            4     1
## 16 Domestic Short Hair, Husky           Dog            4     1
## 17 English Cocker Spaniel, Selkirk Rex  Cat            4     1
## 18 Golden Retriever, LaPerm             Cat            4     1
```

```
## 19 Golden Retriever, Pit Bull Terrier  Dog            4     1
## 20 Labrador Retriever, Silver          Dog            4     1
```




There are 943 of mixed and pure breeds of cats and dogs. There appear to be a few breeds are popular ones and having faster adoption speed than the rest. There are also some breeds that show slower adoption speed.

### 2.4.7 Color

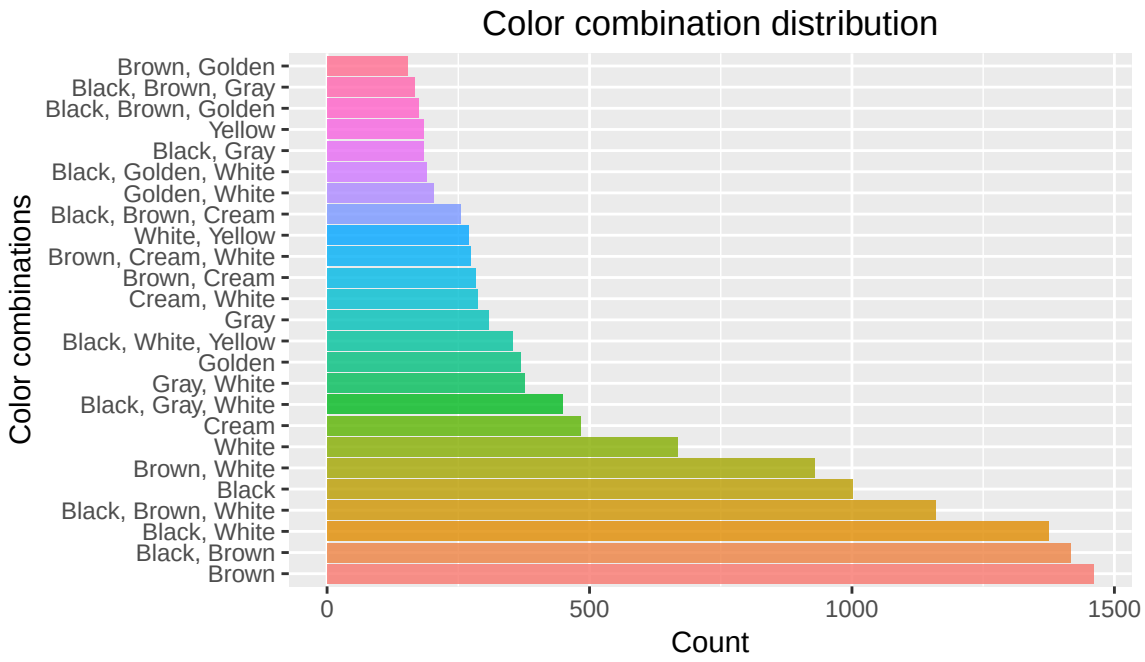
There are 7 color types with respective color labels stated in *df_color* data frame. Pets can be listed as single color whereby the color is indicated in column Color1 with columns Color2 and Color3 having zero value. Mixed colored pets are listed with Color2 and Color3 having values in *df_color* data frame.

```
df_color
```

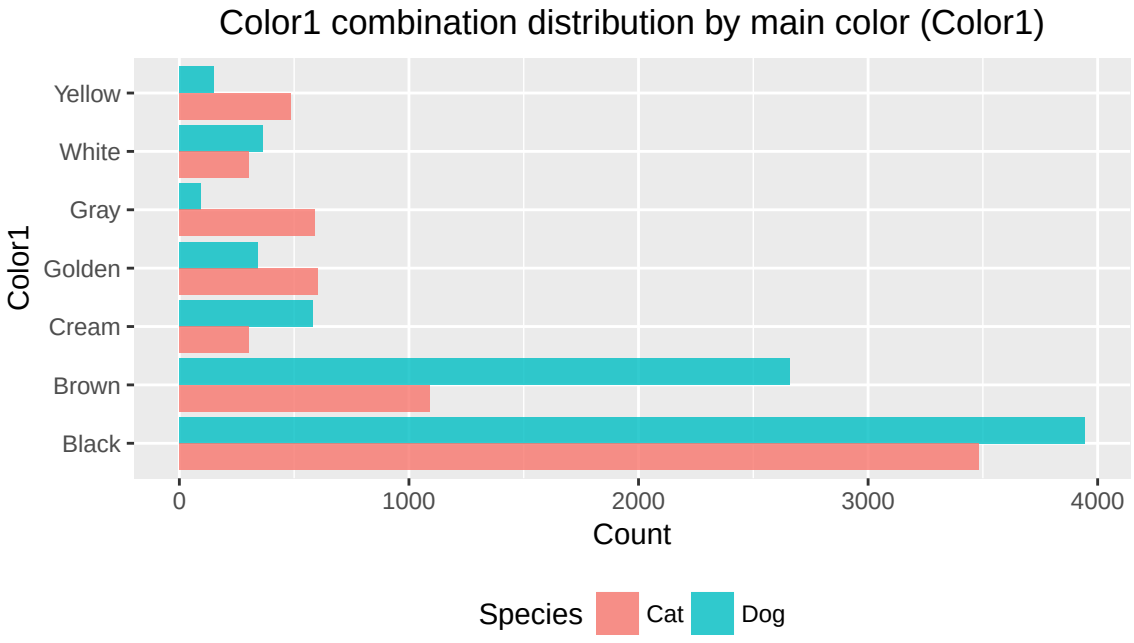
```
##   ColorID ColorName
## 1       1     Black
## 2       2     Brown
## 3       3    Golden
## 4       4    Yellow
## 5       5     Cream
## 6       6      Gray
## 7       7     White
```

Two new columns (Color and Color_Mix) are defined to facilitate data visualization of pet color distributions. However these two columns are not used to build the prediction model.

There are in total 63 unique color combinations.

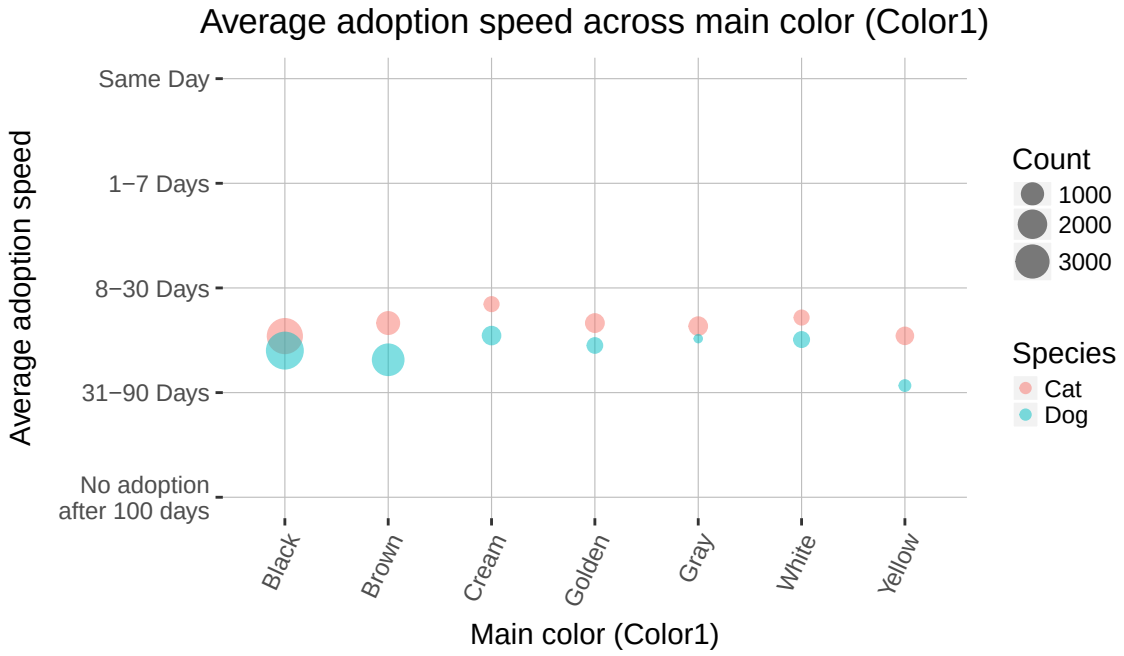



Most pets listed are listed have primary color of pure brown, and mix of black and white. This is followed by pure black and mix of black and brown.



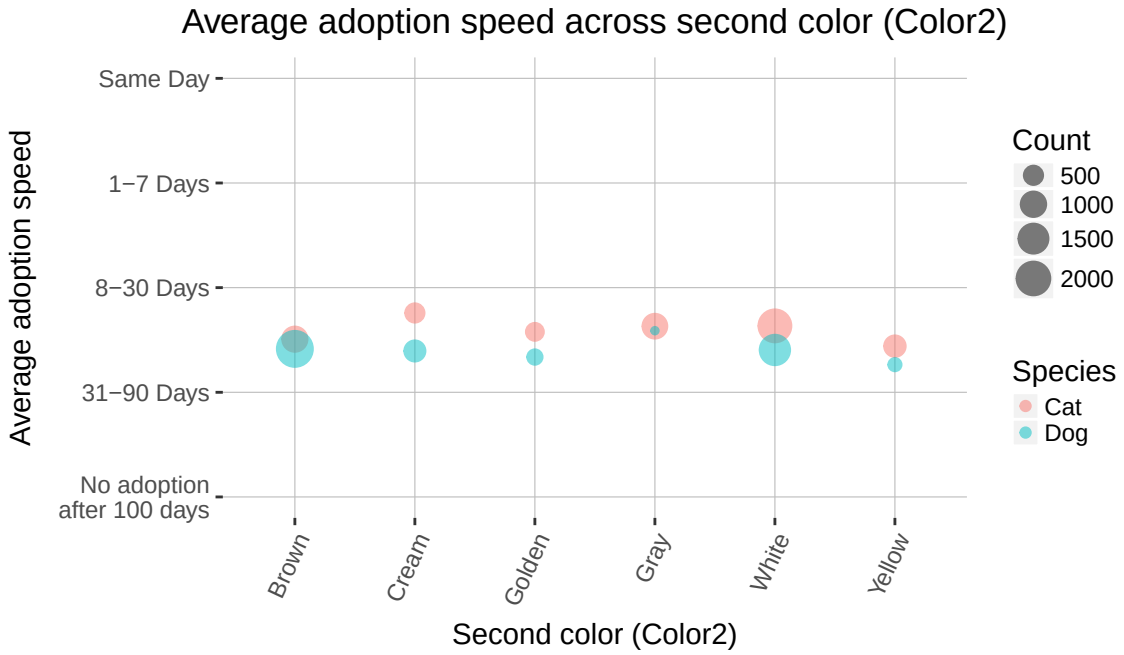

```
## # A tibble: 14 x 4
## # Groups:   Color1 [7]
##    Color1 Species AvgSpeed Count
##    <chr>  <chr>      <dbl> <int>
##  1 Cream  Cat         1.85   304
```

```
##  2 White  Cat          1.72   304
##  3 Brown  Cat          1.66  1091
##  4 Golden Cat          1.66   605
##  5 Gray   Cat          1.63   589
##  6 Cream  Dog          1.54   580
##  7 Yellow Cat          1.54   485
##  8 Black  Cat          1.54  3483
##  9 Gray   Dog          1.52    95
## 10 White  Dog          1.51   363
## 11 Golden Dog          1.45   342
## 12 Black  Dog          1.40  3944
## 13 Brown  Dog          1.31  2659
## 14 Yellow Dog          1.07   149
```

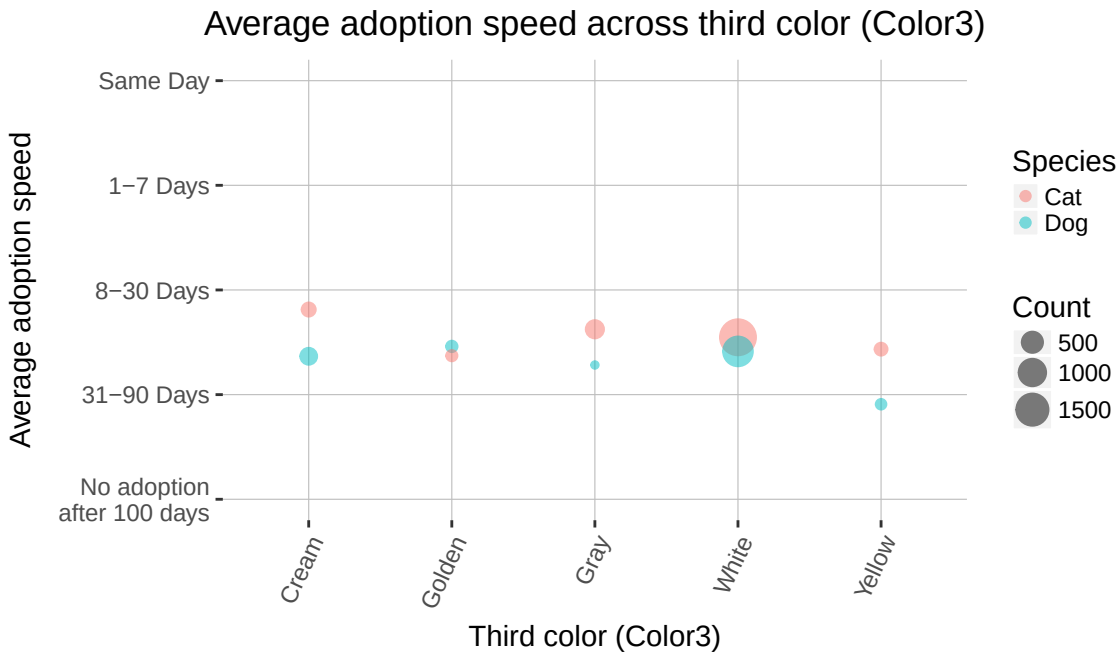



```
## # A tibble: 12 x 4
## # Groups:   Color2 [6]
##    Color2 Species AvgSpeed Count
##    <chr>  <chr>      <dbl> <int>
##  1 Cream  Cat         1.76   500
##  2 White  Cat         1.63  1883
##  3 Gray   Cat         1.63   939
##  4 Gray   Dog         1.59   124
##  5 Golden Cat         1.58   419
##  6 Brown  Cat         1.51   977
##  7 Yellow Cat         1.44   658
##  8 Brown  Dog         1.41  2336
##  9 White  Dog         1.40  1555
```

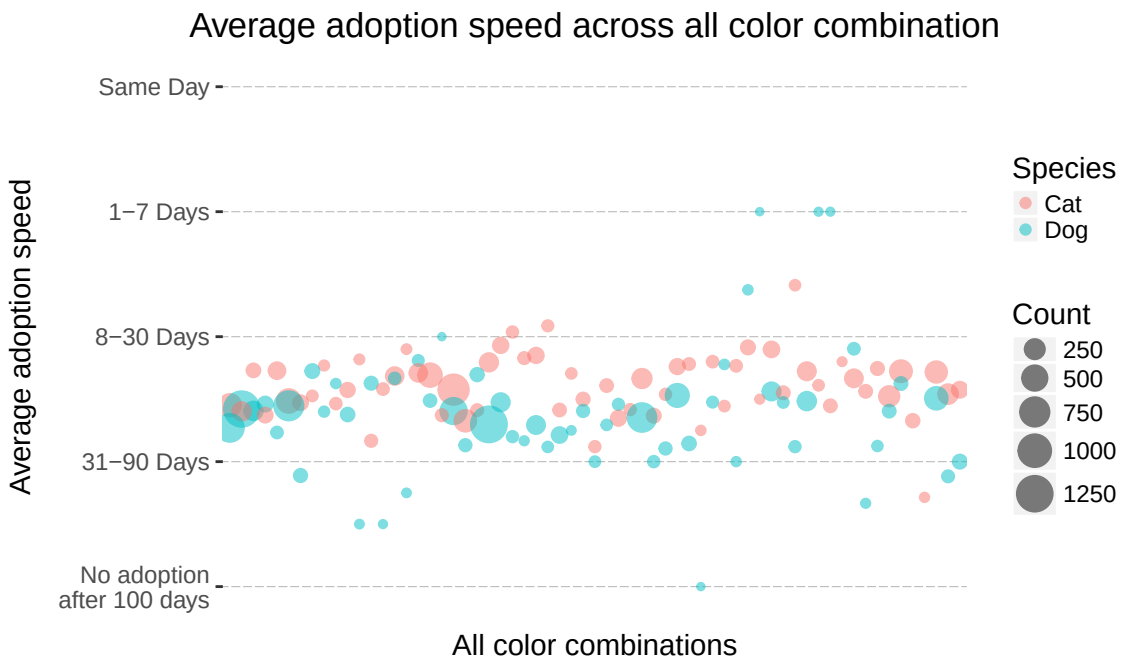
```
## 10 Cream  Dog          1.39   628
## 11 Golden Dog          1.34   291
## 12 Yellow Dog          1.26   212
```



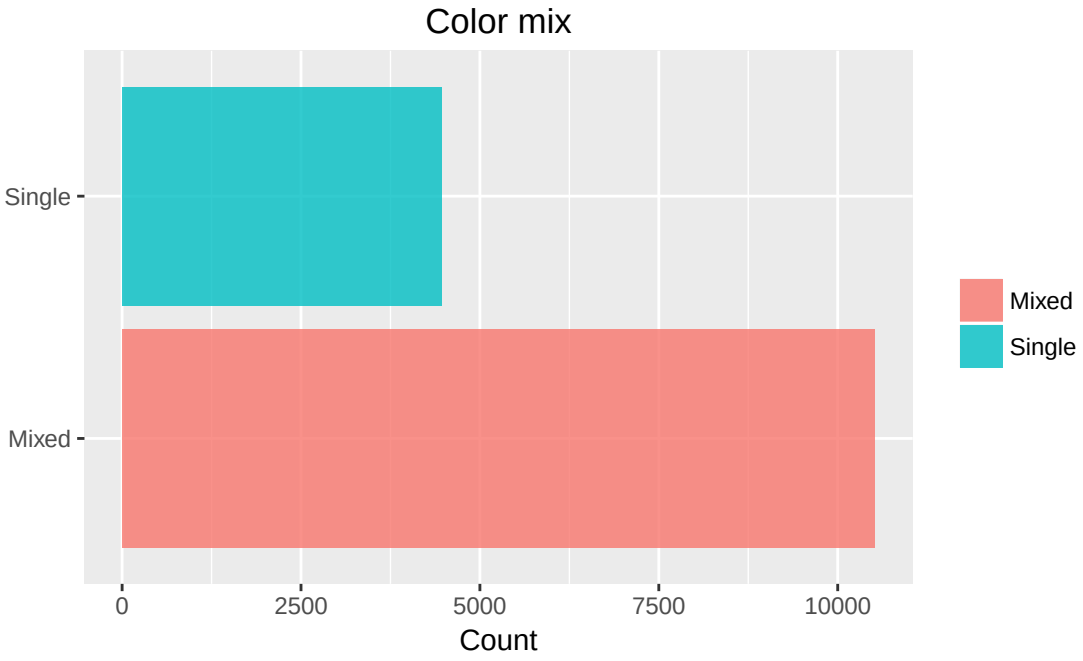

```
## # A tibble: 10 x 4
## # Groups:   Color3 [5]
##    Color3 Species AvgSpeed Count
##    <chr>  <chr>      <dbl> <int>
##  1 Cream  Cat         1.81   155
##  2 Gray   Cat         1.62   325
##  3 White  Cat         1.55  1991
##  4 Golden Dog         1.46    89
##  5 Yellow Cat         1.43   122
##  6 White  Dog         1.41  1230
##  7 Golden Cat         1.37    86
##  8 Cream  Dog         1.37   262
##  9 Gray   Dog         1.28    53
## 10 Yellow Dog        0.908    76
```




```
## # A tibble: 20 x 4
## # Groups:   Color_All [19]
##    Color_All             Species AvgSpeed Count
##    <chr>                 <chr>      <dbl> <int>
##  1 Cream, Gray, Yellow   Dog            3     1
##  2 Golden, Gray          Dog            3     2
##  3 Golden, Gray, White   Dog            3     2
##  4 Cream, Yellow         Cat         2.41    17
##  5 Cream, Gray, White    Dog         2.38     8
##  6 Brown, Cream, Yellow  Cat         2.09    23
##  7 Brown, Cream, Golden  Cat         2.04    27
##  8 Black, Gray, Yellow   Dog         2        1
##  9 Brown, Cream          Cat         1.93   101
## 10 Cream, Gray, White    Cat         1.91    70
## 11 Golden, White         Dog         1.90    31
## 12 Black, Golden, Yellow Cat         1.9     10
## 13 Cream, White          Cat         1.90   108
## 14 Brown, Cream, White   Cat         1.85   100
## 15 Brown, Cream, Gray    Cat         1.83    35
## 16 Black, Cream, Yellow  Cat         1.82    11
## 17 Black, Gray           Dog         1.81    21
## 18 Cream, Golden, White  Cat         1.8     30
## 19 Golden, Gray, Yellow  Cat         1.8      5
## 20 Brown                 Cat         1.79   185
```

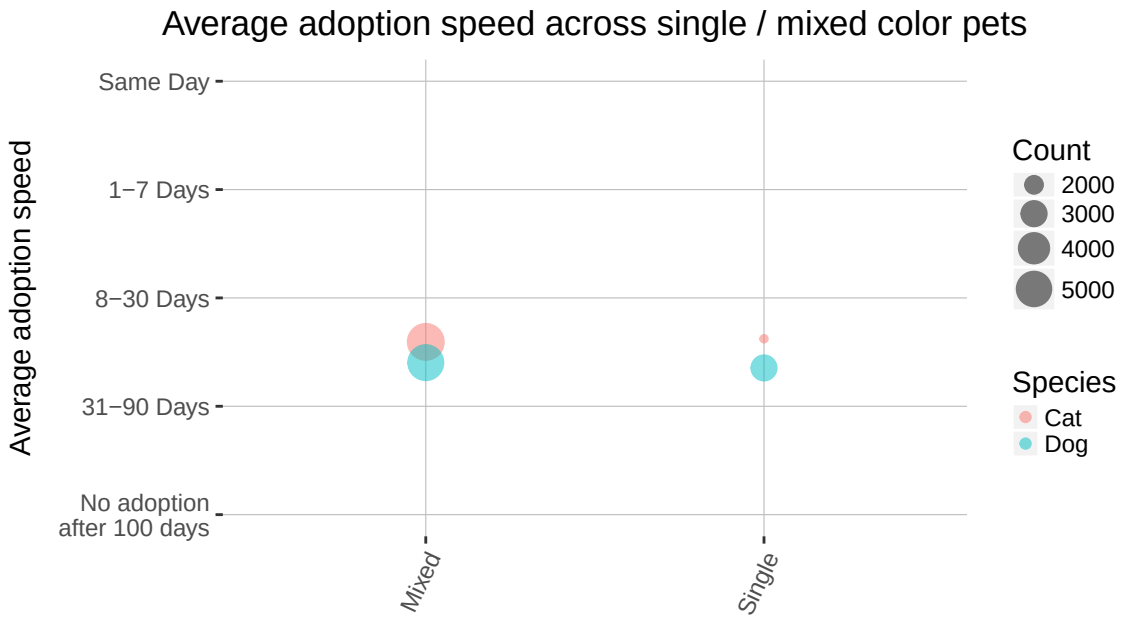



The following plot shows the distribution of single and mixed colored pets. Pets listed with mixed colors more than double of the single colored ones.




```
## # A tibble: 4 x 4
## # Groups:   Color_Mix [2]
##   Color_Mix Species AvgSpeed Count
##   <chr>     <chr>      <dbl> <int>
```

```
## 1 Single     Cat          1.62  1485
## 2 Mixed      Cat          1.59  5376
## 3 Mixed      Dog          1.40  5146
## 4 Single     Dog          1.35  2986
```




The distribution of average adoption speed of pets having primary color (Color1) appear to have similar average adoption speed. The same observation holds for second color (Color2) and third color (Color3). That is also the case when all colors are combined. Most color combinations somewhat have similar average adoption speed, with some random small number of pets being adopted faster and slower. Pet color does not seem to affect the adoption speed.

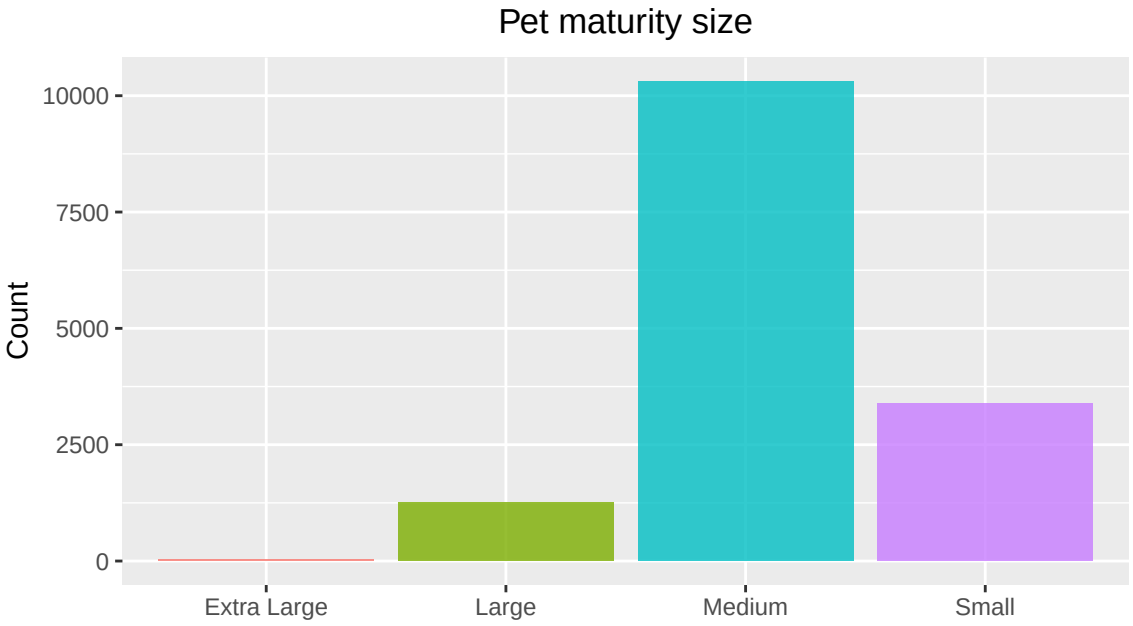
### 2.4.8 Maturity size

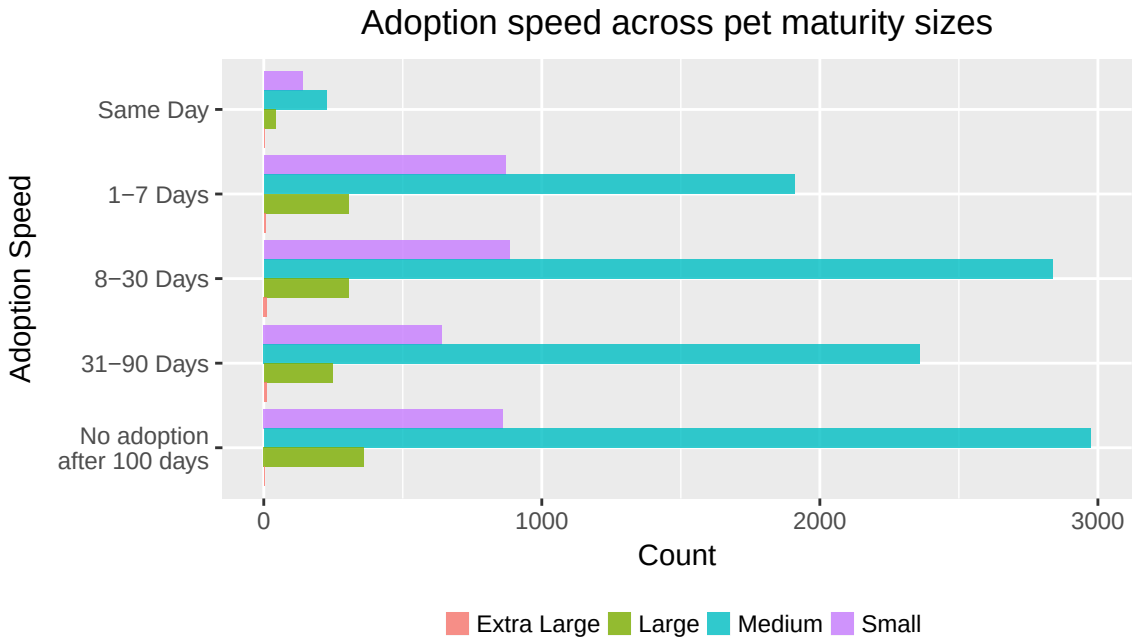
Maturity size of pets in this data set ranges from *small*, *medium*, *large* up to *extra large* indicated with values from 1 to 4. The data set documentation states pets with unspecified maturity size are indicated with value 0. A quick check on the data set shows that all pets listed has maturity size specified.

```
nrow(df_pets %>% filter(MaturitySize==0))
```

```
## [1] 0
```

Pets listed for adoption are mostly those with maturity size of *medium*, followed by *small*. The adoption speed for different maturity sizes shows the same distribution which could be due to the distribution of pet numbers in each maturity size categories. The average adoption speed across the maturity sizes show that *extra large* dogs have faster adoption speed on average. However, there are not many such dogs in the sample.

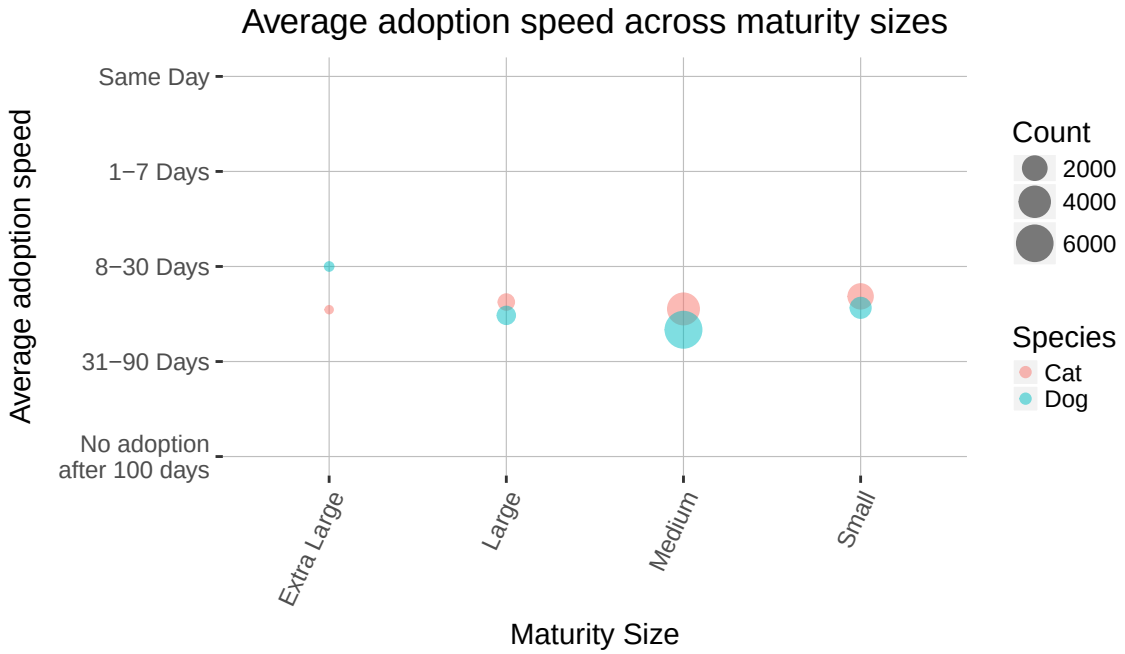



```
## Warning: Grouping rowwise data frame strips rowwise nature
```

```
## # A tibble: 6 x 4
## # Groups:   MaturitySize [4]
##   MaturitySize Species AvgSpeed Count
##   <chr>        <chr>      <dbl> <int>
## 1 Extra Large  Dog          2       22
## 2 Small        Cat          1.69  2172
## 3 Large        Cat          1.63   500
```

```
## 4 Small       Dog         1.57  1223
## 5 Medium      Cat         1.55  4178
## 6 Extra Large Cat         1.55    11
```



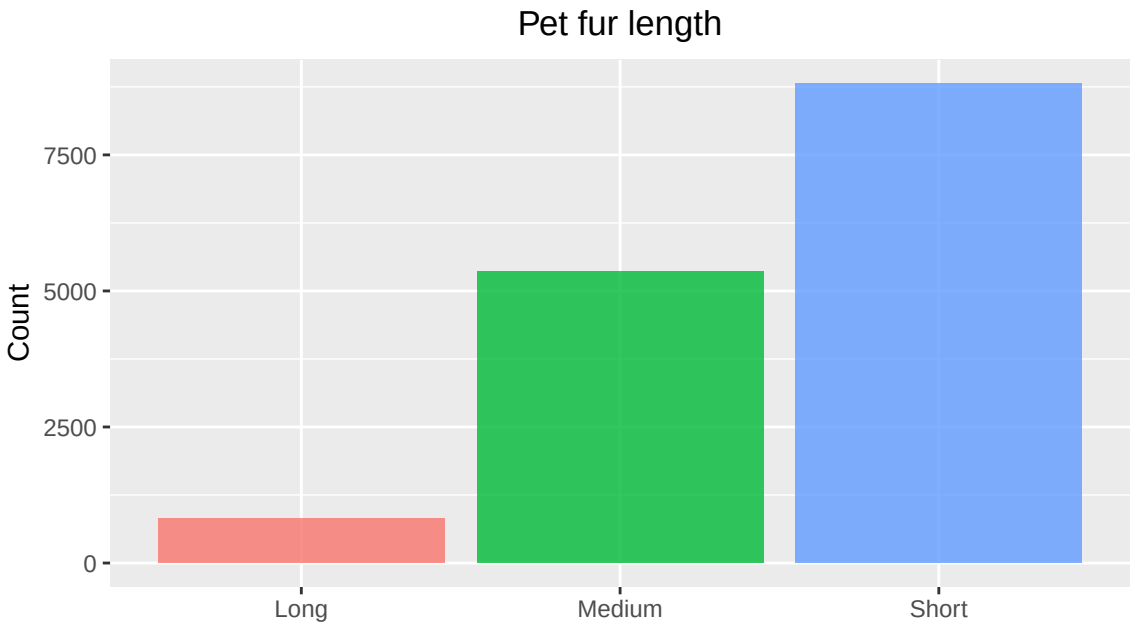

### 2.4.9 Fur length

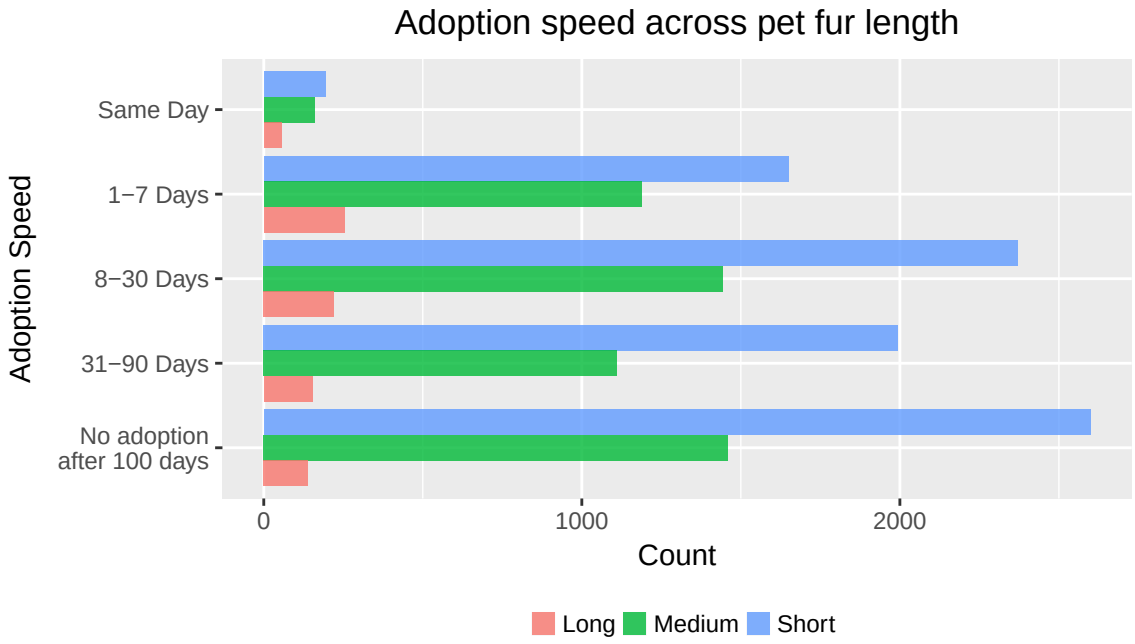
Fur length of pets in this data set ranges from *short*, *medium* up to *long* indicated with values from 1 to 3. The data set documentation states pets with unspecified fur length are indicated with value 0. A quick check on the data set shows that all pets listed has fur length specified.

```
nrow(df_pets %>% filter(FurLength==0))
```

```
## [1] 0
```

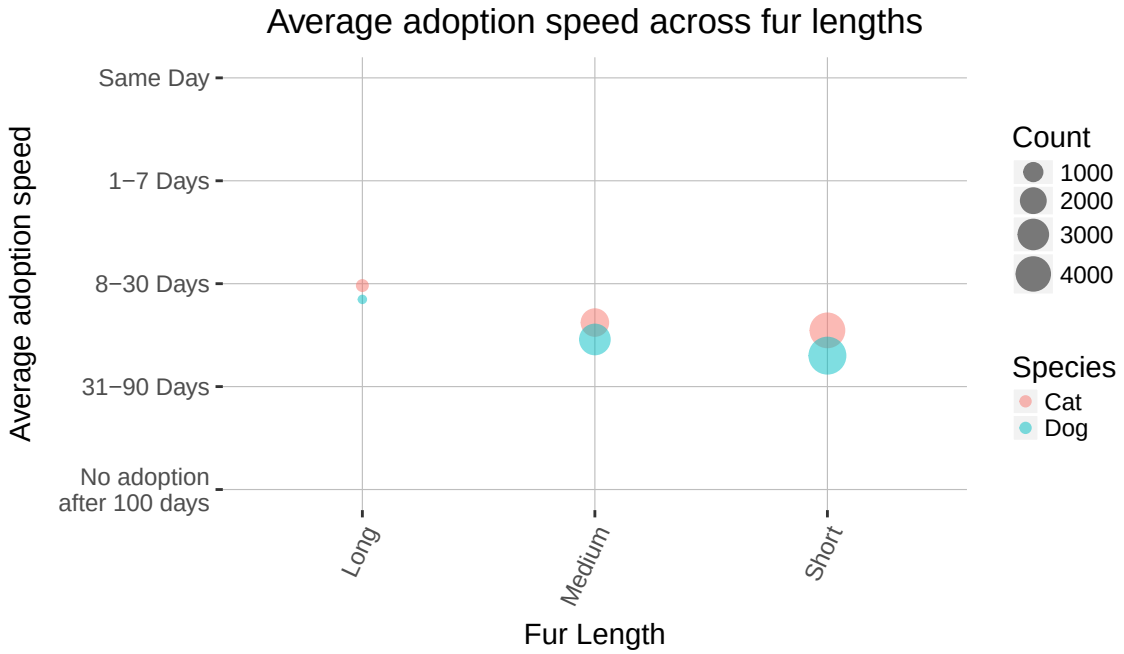
Most pets listed for adopted has *short* fur, followed by pets with *medium* fur length. It is not surprising to see the same distribution of adoption speed to show the same. The average adoption speed across the fur lengths show that cats and dogs with *long* furs have faster adoption speed on average.



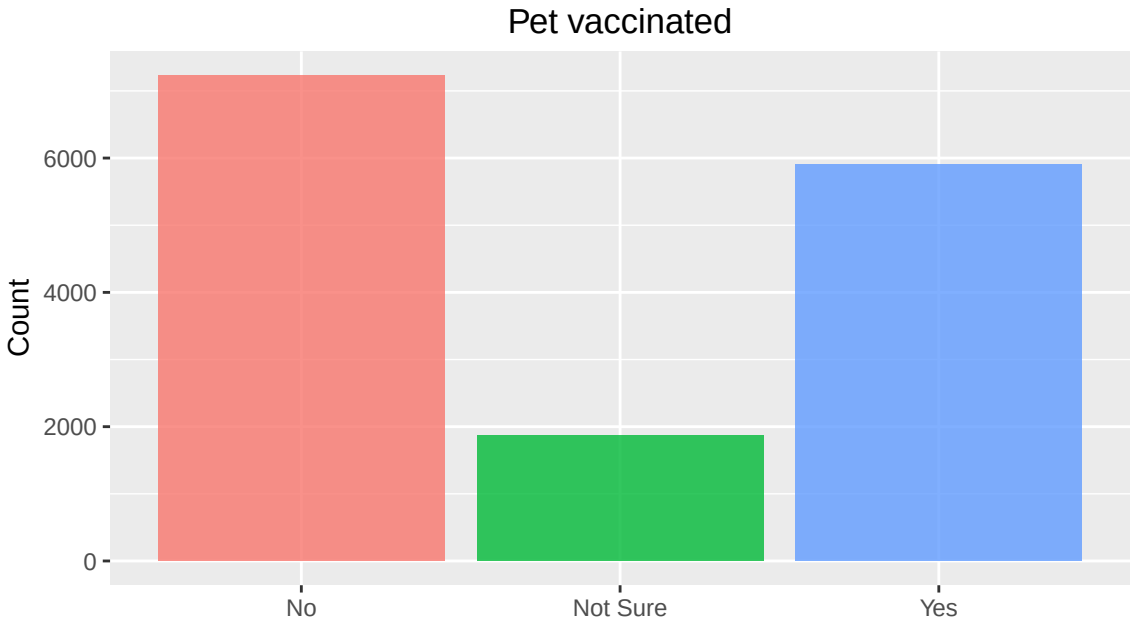

```
## # A tibble: 6 x 4
## # Groups:   FurLength [3]
##   FurLength Species AvgSpeed Count
##   <chr>     <chr>      <dbl> <int>
## 1 Long      Cat         1.98   444
## 2 Long      Dog         1.85   380
## 3 Medium    Cat         1.62  2333
## 4 Short     Cat         1.55  4084
```

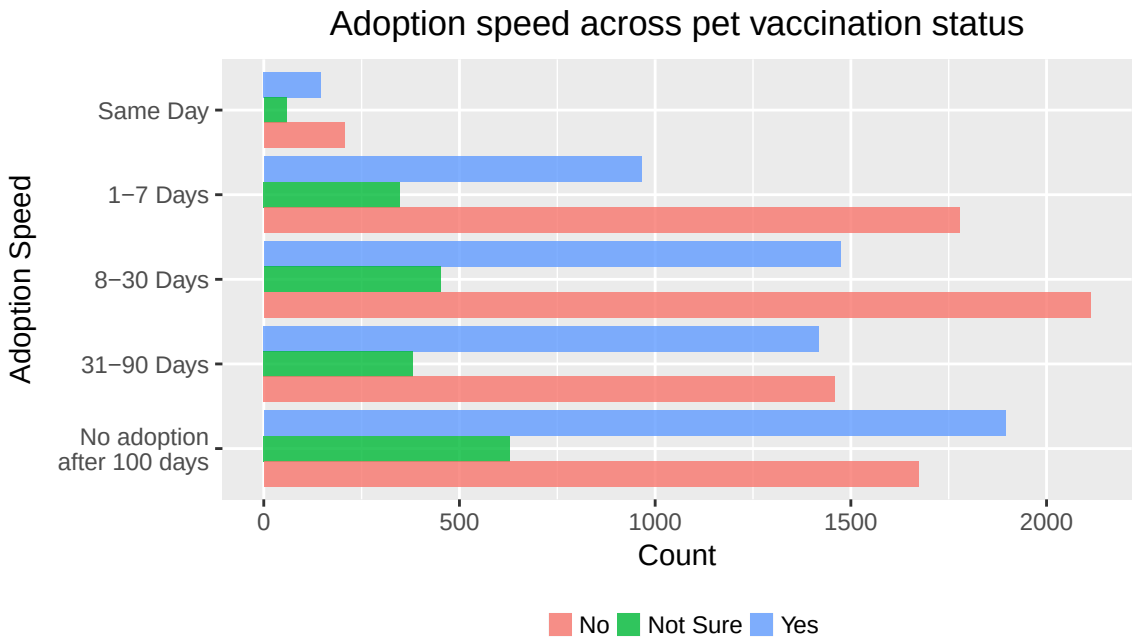
```
## 5 Medium     Dog          1.46  3028
## 6 Short      Dog          1.30  4724
```



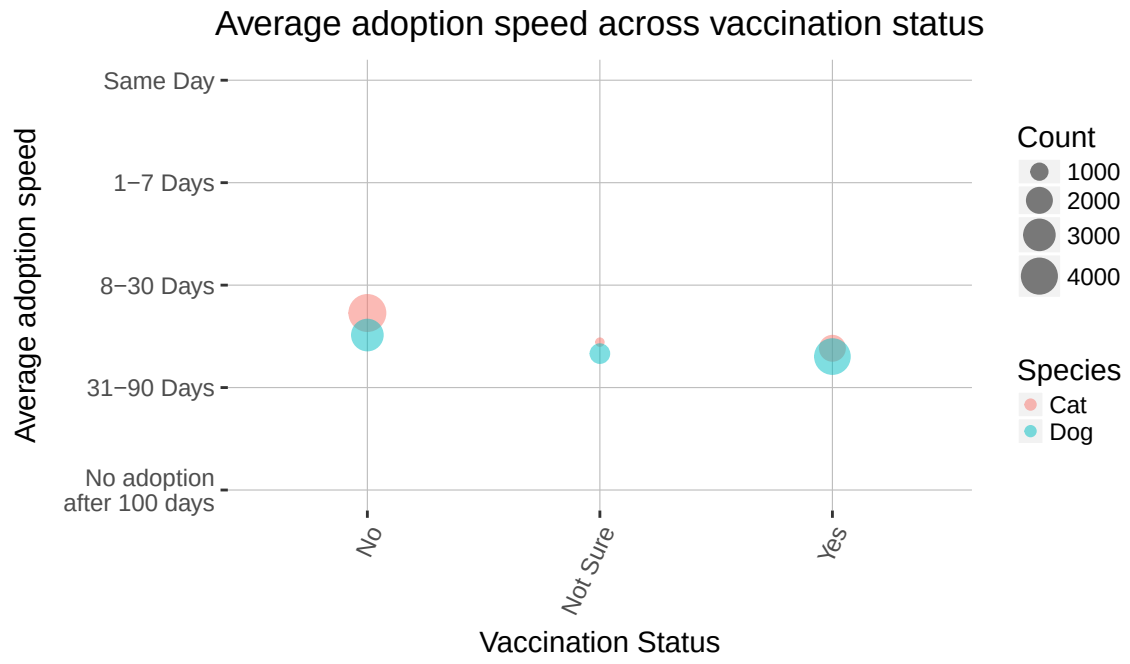

### 2.4.10 Vaccination status

Vaccination status of pets in this data set can be *Yes*, *No* or *Not Sure* with values from 1 to 3.





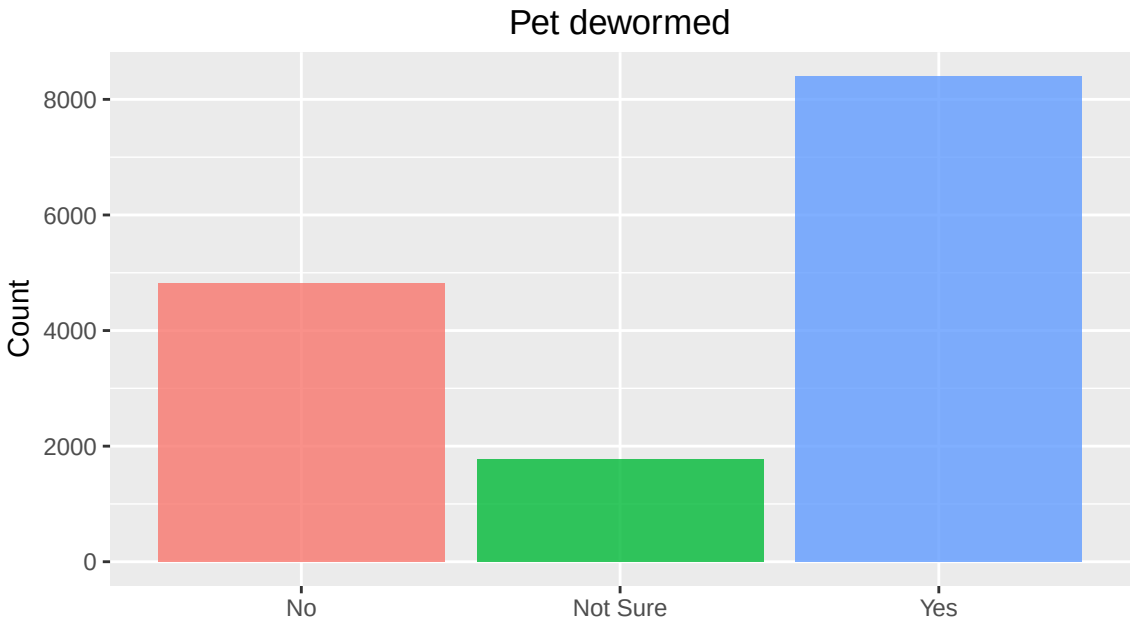

```
## # A tibble: 6 x 4
## # Groups:   Vaccinated [3]
##   Vaccinated Species AvgSpeed Count
##   <chr>      <chr>      <dbl> <int>
## 1 No         Cat         1.73  4208
## 2 No         Dog         1.51  3019
## 3 Not Sure   Cat         1.44   661
## 4 Yes        Cat         1.38  1992
## 5 Not Sure   Dog         1.33  1207
## 6 Yes        Dog         1.30  3906
```

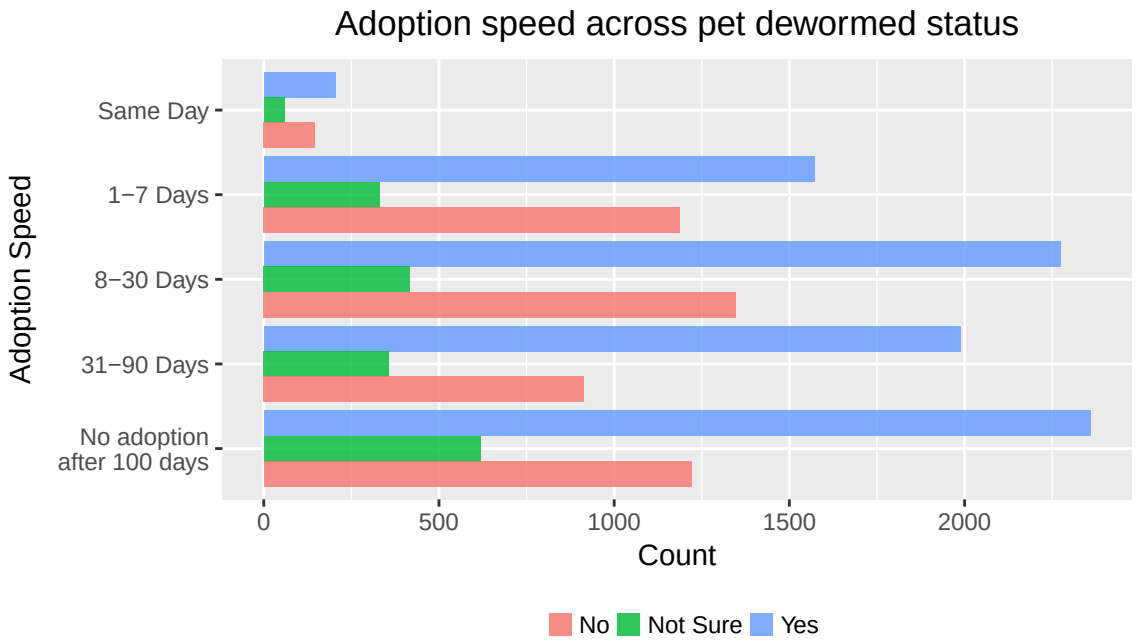



Most pets appear to be not vaccinated. However, the numbers of vaccinated pets are close to the those not vaccinated. The adoption speed for different vaccination status shows the same distribution which could be due to the distribution of pet numbers in each vaccination status categories. Average adoption speeds for pets appear to be similar across different vaccination status. Perhaps people do not bother, and would vaccinate the pets if it is not vaccinated or unsure of the status.

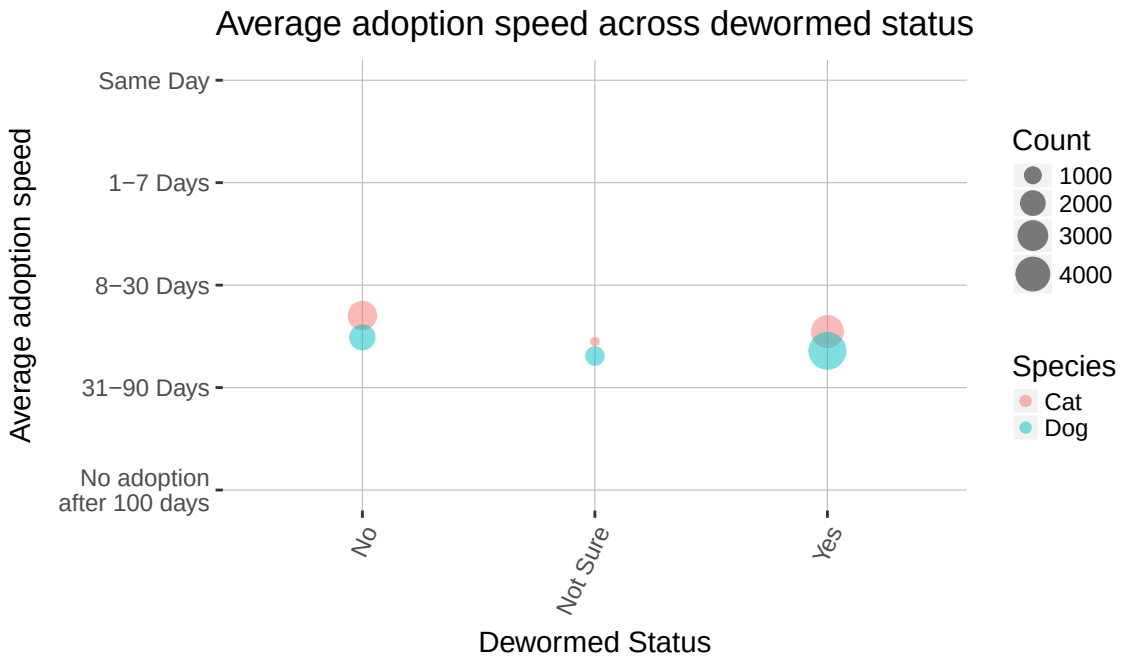
### 2.4.11 Deworm status

Deworming status of pets in this data set can be *Yes*, *No* or *Not Sure* with values from 1 to 3.





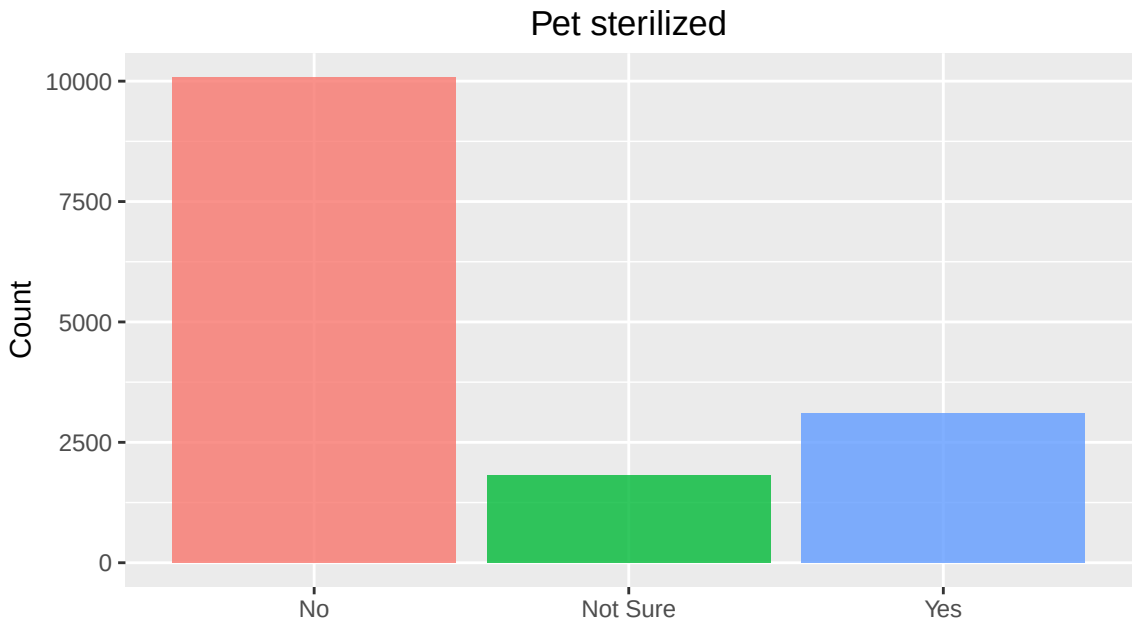

```
## # A tibble: 6 x 4
## # Groups:   Dewormed [3]
##   Dewormed Species AvgSpeed Count
##   <chr>    <chr>      <dbl> <int>
## 1 No       Cat         1.70  2734
## 2 Yes      Cat         1.55  3511
## 3 No       Dog         1.49  2081
## 4 Not Sure Cat         1.45   616
## 5 Yes      Dog         1.36  4886
## 6 Not Sure Dog         1.31  1165
```

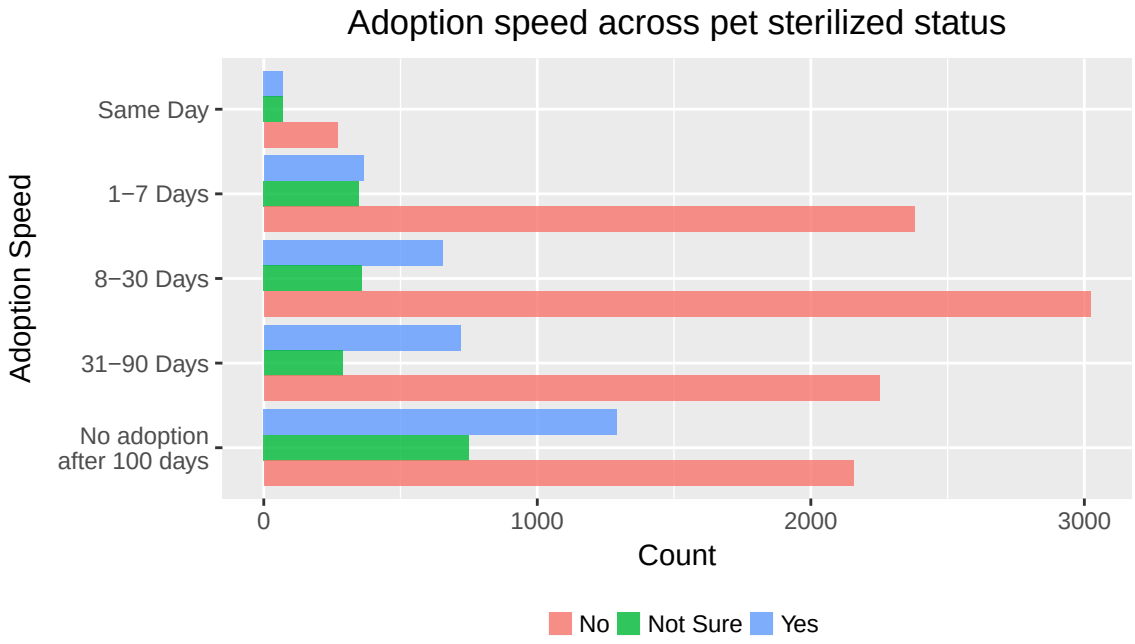



Most pets are listed appear to be dewormed. However the combined number of pets with dewormed status *No* and *Not Sure* is close to pets having *Yes* dewormed status. The adoption speed for different dewormed status shows the same distribution which could be due to the distribution of pet numbers in each dewormed status categories. Average adoption speeds for pets appear to be similar across different dewormed status.

### 2.4.12 Sterilization status

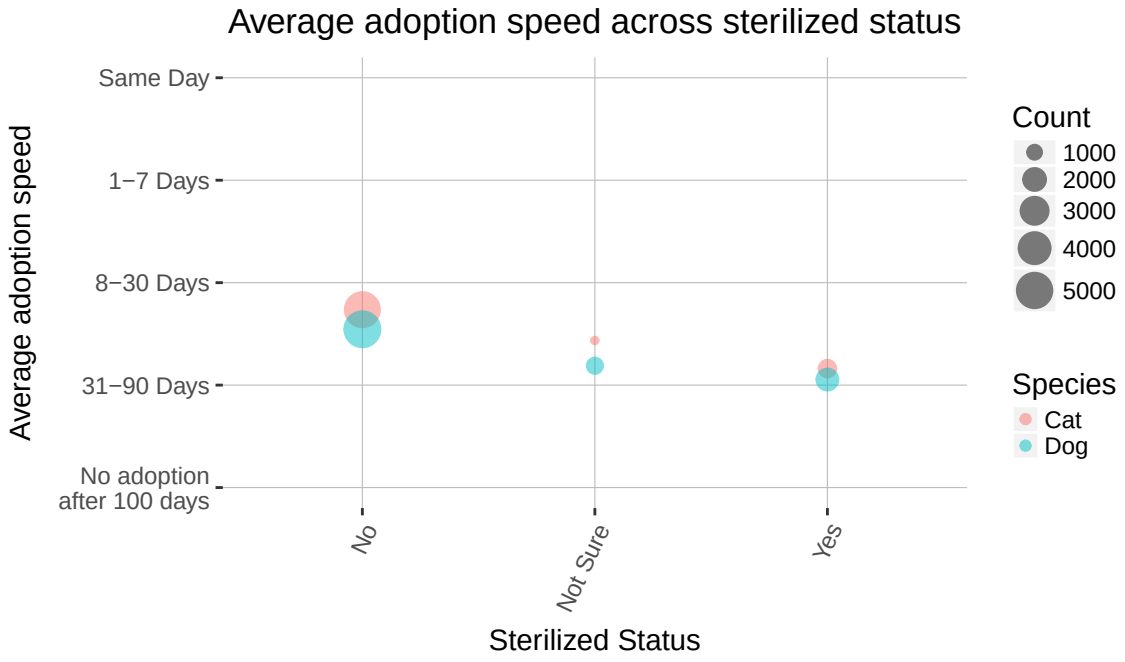
Sterilization (spayed or neutered) status of pets in this data set can be *Yes*, *No* or *Not Sure* with values from 1 to 3.




```
## # A tibble: 6 x 4
## # Groups:   Sterilized [3]
##   Sterilized Species AvgSpeed Count
##   <chr>      <chr>      <dbl> <int>
## 1 No         Cat         1.74  4905
## 2 No         Dog         1.55  5172
## 3 Not Sure   Cat         1.44   698
## 4 Not Sure   Dog         1.19  1117
```

```
## 5 Yes         Cat          1.16  1258
## 6 Yes         Dog          1.05  1843
```



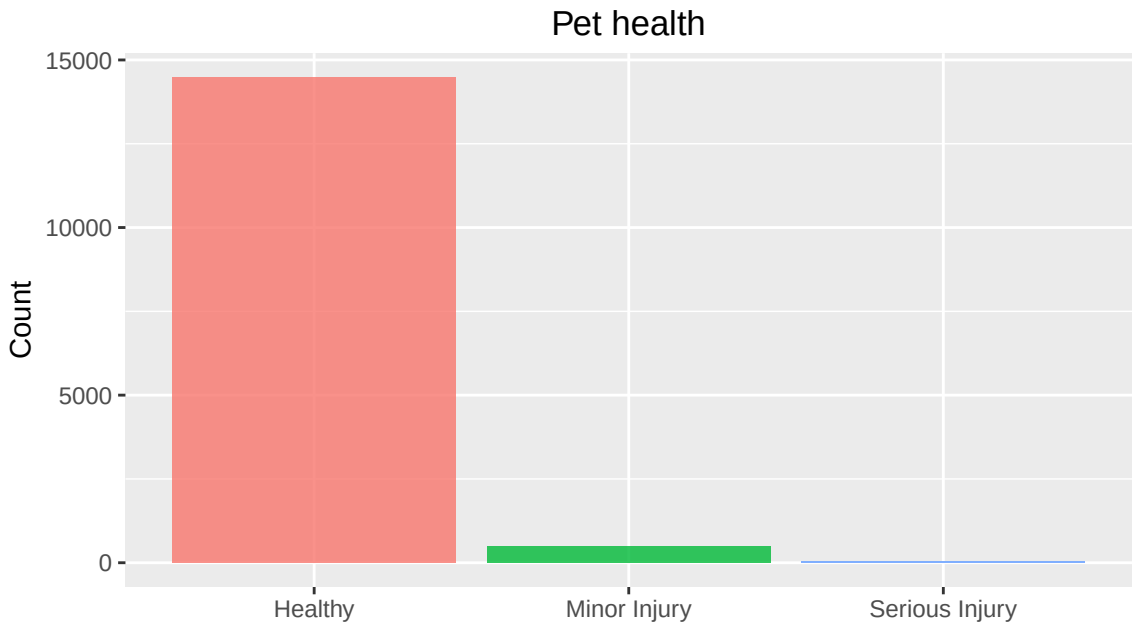

Most pets are listed as not sterilized. The adoption speed for different sterilization status shows the same distribution which could be due to the distribution of pet numbers in each sterilization status categories. Average adoption speeds show pets that are not sterilized have faster adoption speed.

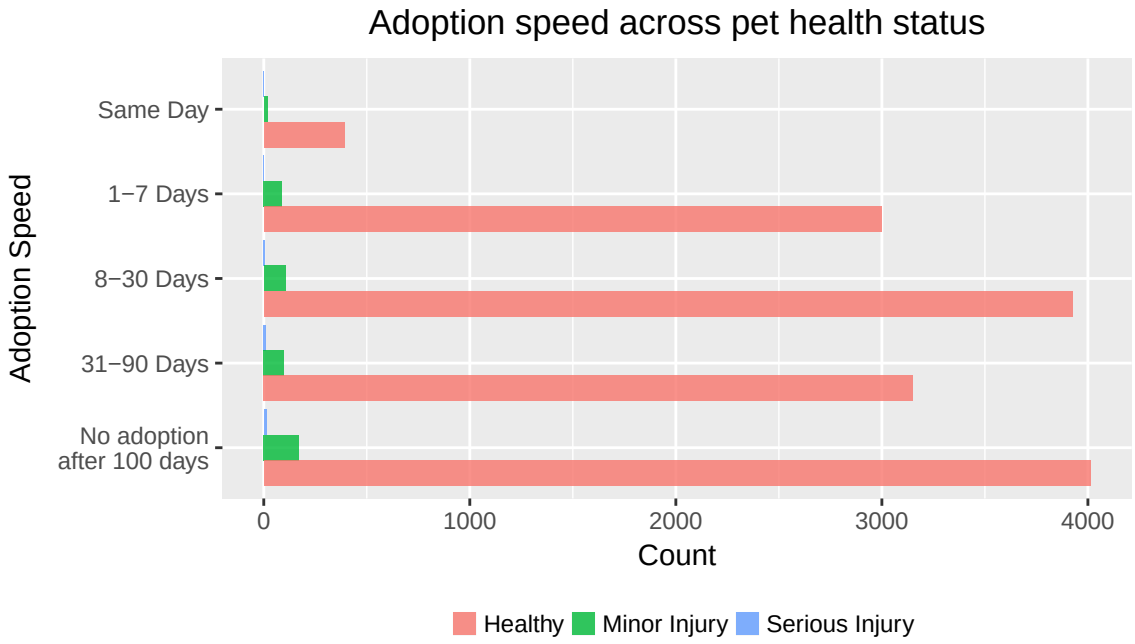
### 2.4.13 Health status

Health status of pets in this data set can be listed as *Healthy*, *Minor Injury* or *Serious Injury* with values from 1 to 3. The data set documentation states pets with unspecified health status are indicated with value 0. A quick check on the data set shows that all pets listed has health status specified.

```
nrow(df_pets %>% filter(Health==0))
```

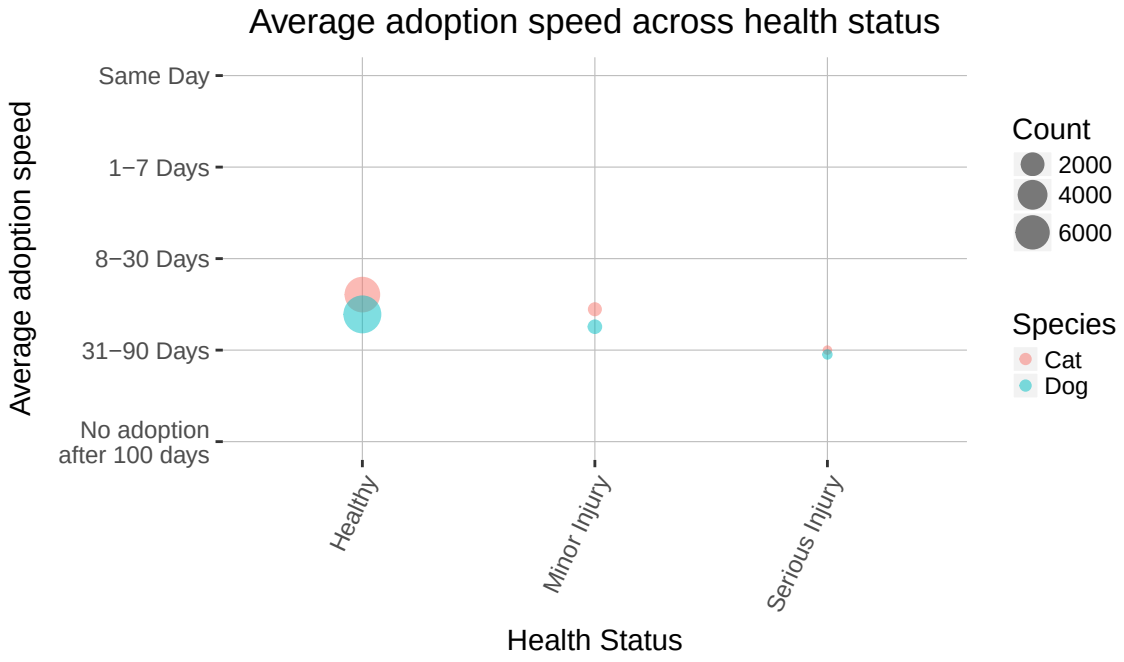
```
## [1] 0
```




```
## # A tibble: 6 x 4
## # Groups:   Health [3]
##   Health       Species AvgSpeed Count
##   <chr>        <chr>      <dbl> <int>
## 1 Healthy      Cat         1.61  6633
## 2 Minor Injury Cat         1.45   215
## 3 Healthy      Dog         1.39  7845
## 4 Minor Injury Dog         1.26   266
```

```
## 5 Serious Injury Cat         1         13
## 6 Serious Injury Dog         0.952     21
```




Almost all pets are listed as *healthy*. Perhaps the person who list the pet wanted to promote people to take up the pet? The adoption speed for different health status shows the same distribution which could be due to the distribution of pet numbers in each health status categories. Average adoption speeds show pets that are healthy have faster adoption speed, while those with *serious injury* has slower adoption speed.

### 2.4.14 Pet quantity

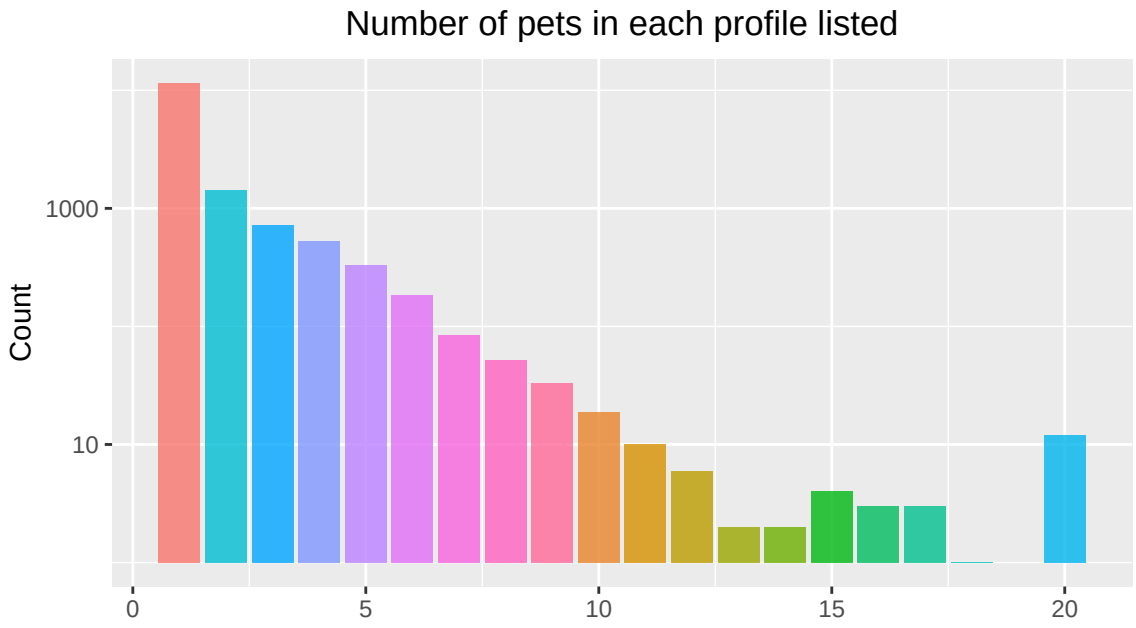
The number of pets represented in each profile listing is indicated in the column Quantity. A quick check of the data set indicate there is no profile listed with Quantity value of less than 0. So the data set is clean in this aspect.

```
nrow(df_pets %>% filter(Quantity<=0))
```

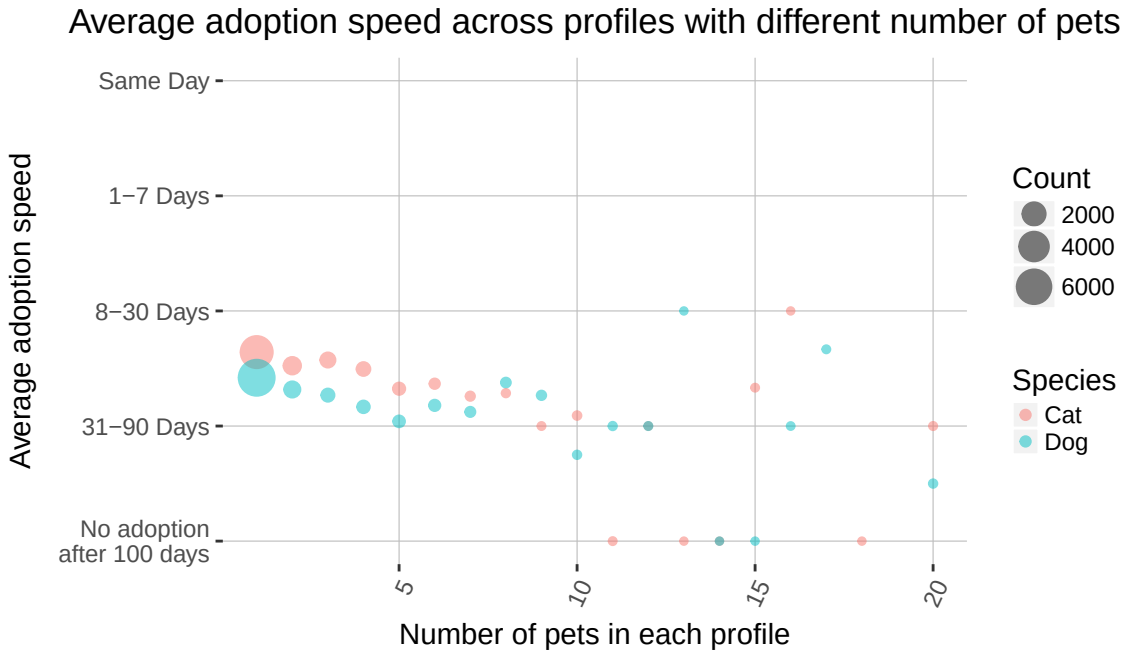
```
## [1] 0
```

```
## # A tibble: 19 x 2
##    Quantity Count
##    <chr>    <int>
##  1 1        11565
##  2 10          19
##  3 11          10
##  4 12           6
##  5 13           2
##  6 14           2
```

```
##  7 15           4
##  8 16           3
##  9 17           3
## 10 18           1
## 11 2         1422
## 12 20          12
## 13 3          726
## 14 4          531
## 15 5          333
## 16 6          185
## 17 7           84
## 18 8           52
## 19 9           33
```




```
## # A tibble: 36 x 4
## # Groups:   Quantity [19]
##    Quantity Species AvgSpeed Count
##    <chr>    <chr>      <dbl> <int>
##  1 13       Dog            2     1
##  2 16       Cat            2     1
##  3 17       Dog            1.67  3
##  4 1        Cat            1.64  4943
##  5 3        Cat            1.57  469
##  6 2        Cat            1.52  811
##  7 4        Cat            1.50  321
##  8 1        Dog            1.42  6622
##  9 8        Dog            1.38  45
## 10 6        Cat            1.37  68
## # ... with 26 more rows
```




Each profile usually lists less than 5 pets. However, there are some with quantity value as high as 20. Adoption speed of each profile is based on all the pets in the profile get adopted. Hence, profiles with large number of pets would take more time to be completed adopted.

Average adoption speeds show profiles with less number of pets have faster adoption speed, while those with more number of pets has slower adoption speed, though there are few exceptional cases.

### 2.4.15 Adoption fee

The adoption fee is in the column Fee, with 0 indicating the pet is available for free. A quick check of the data set indicate there is no profile listed with Fee value of less than 0. So the data set is clean in this aspect.

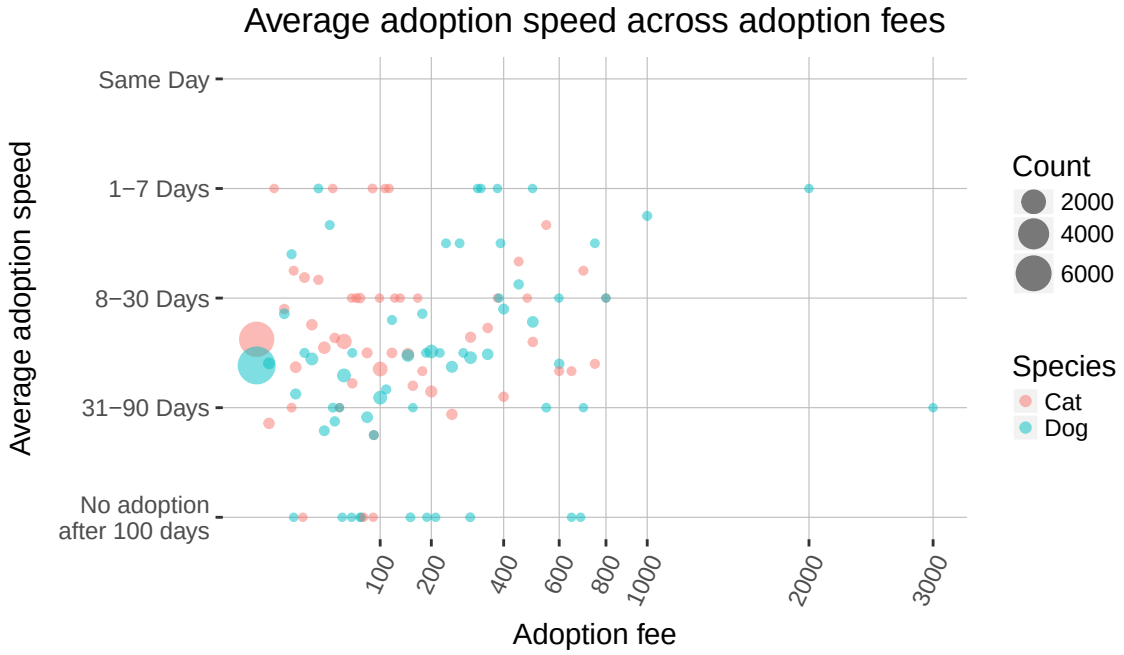
```
nrow(df_pets %>% filter(Fee <0))
```

```
## [1] 0
```

```
## # A tibble: 74 x 3
##      Fee Count   Average
##    <dbl> <int>     <dbl>
##  1     0 12663 0.845
##  2     1    82 0.00547
##  3     2     1 0.0000667
##  4     5    24 0.00160
##  5     8     7 0.000467
##  6     9     5 0.000333
##  7    10    70 0.00467
##  8    14     1 0.0000667
```

```
##  9    15    20 0.00133
## 10    20   136 0.00907
## # ... with 64 more rows
```

```
## # A tibble: 112 x 4
## # Groups:   Fee [74]
##      Fee Species AvgSpeed Count
##    <dbl> <chr>      <dbl> <int>
##  1     2 Cat            3     1
##  2    25 Dog            3     2
##  3    38 Cat            3     1
##  4    88 Cat            3     2
##  5   108 Cat            3     1
##  6   115 Cat            3     1
##  7   320 Dog            3     1
##  8   330 Dog            3     1
##  9   380 Dog            3     1
## 10   499 Dog            3     1
## # ... with 102 more rows
```

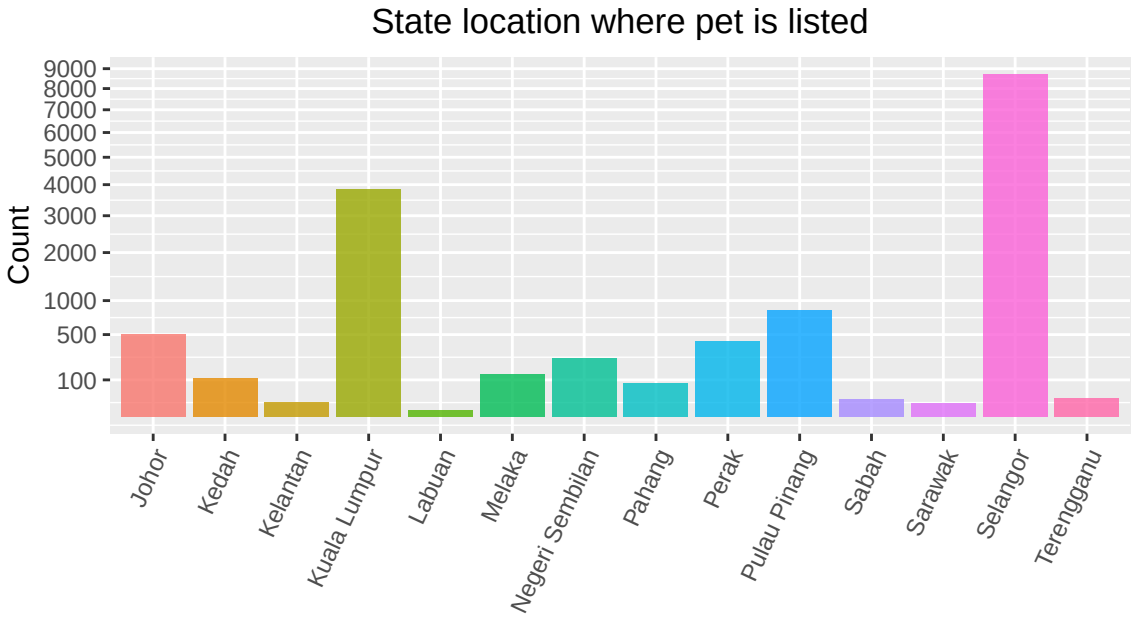



84% of pets are listed as free, with others having fees up to 3000. It is interesting to note that pets listed for free fall somewhere in between the adoption speeds. However, there are people who are willing to pay up to 2000 and adopt the pet within a week of its listing. Similarly, there are pets with adoption fee that end up not adopted after 100 days. Perhaps people are willing to pay a fee for a healthy pet, particular breed or other characteristics.

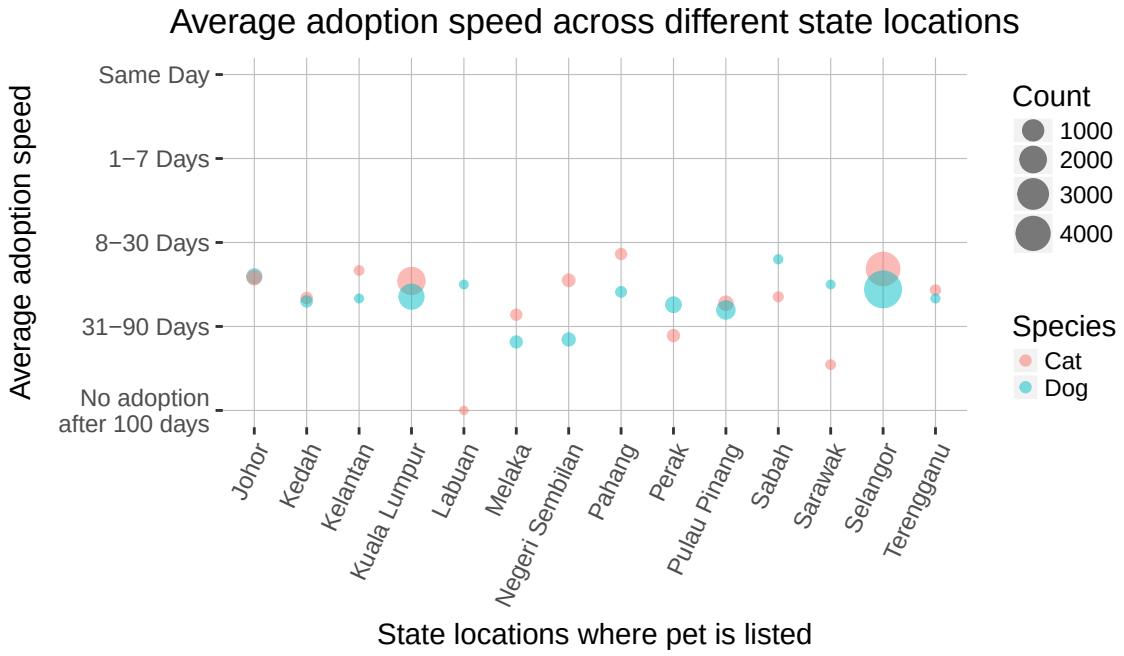
### 2.4.16 State locations

Some states have more pets listed for adoption than the rest. Average adoption speeds for pets appear to be similar across different state locations, with a few states where pets with small numbers are adopted at faster or slower adoption speed than average.

```
## [1] 0
```



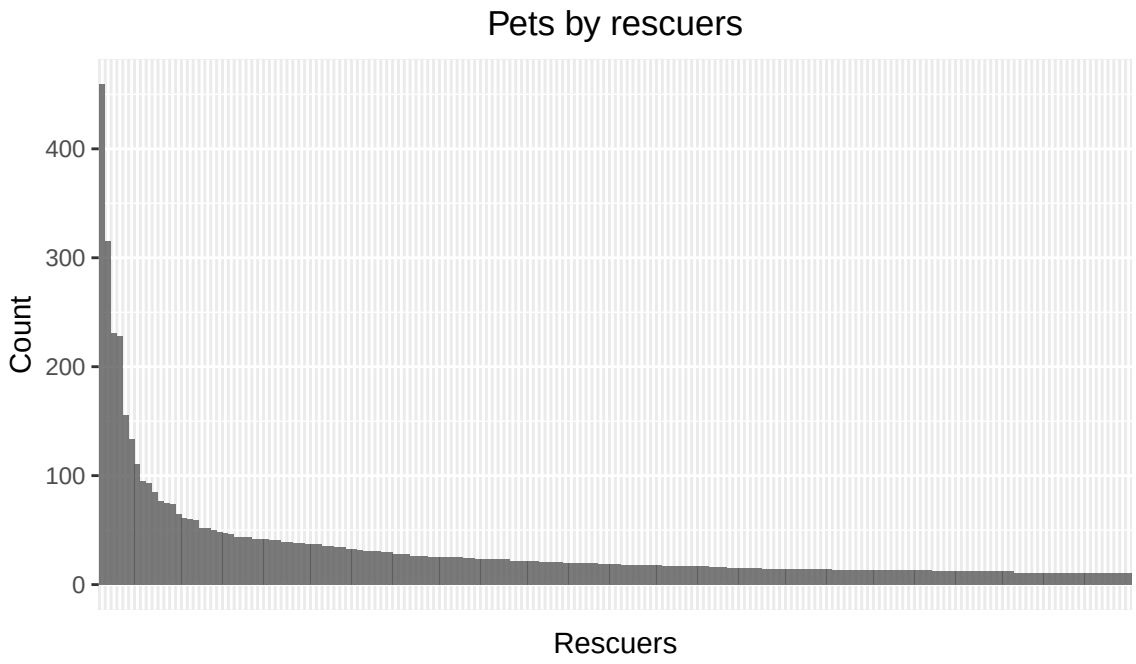

```
## # A tibble: 6 x 4
## # Groups:   StateName [5]
##   StateName Species AvgSpeed Count
##   <fct>     <chr>      <dbl> <int>
## 1 Pahang    Cat         1.86    51
## 2 Sabah     Dog         1.8      5
## 3 Selangor  Cat         1.68  3821
## 4 Kelantan  Cat         1.67    12
## 5 Johor     Dog         1.60   278
## 6 Johor     Cat         1.58   229
```



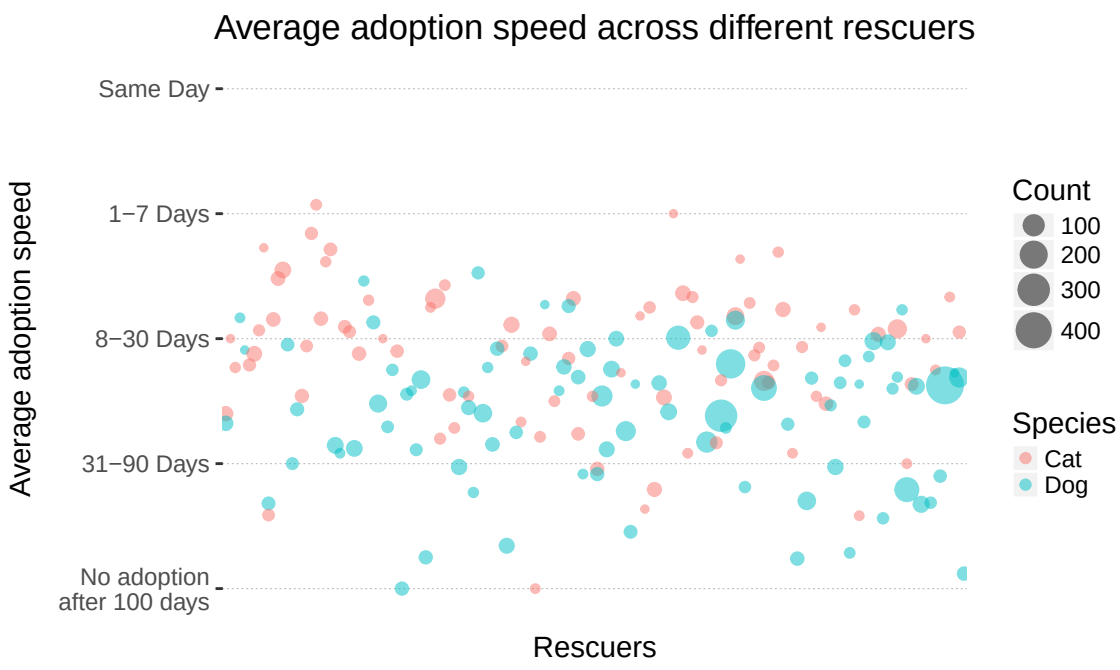

### 2.4.17 Pet rescuers

ResuerID in this data set is masked. However, the data set preserves a unique mask to uniquely identify the rescuers.

```
## # A tibble: 5,595 x 2
##    RescuerID                           n
##    <fct>                           <int>
##  1 fa90fa5b1ee11c86938398b60abc32cb   459
##  2 aa66486163b6cbc25ea62a34b11c9b91   315
##  3 c00756f2bdd8fa88fc9f07a8309f7d5d   231
##  4 b53c34474d9e24574bcec6a3d3306a0d   228
##  5 ee2747ce26468ec44c7194e7d1d9dad9   156
##  6 95481e953f8aed9ec3d16fc4509537e8   134
##  7 b770bac0ca797cf1433c48a35d30c4cb   111
##  8 a042471e0f43f2cf707104a1a138a7df    95
##  9 fd970cc91d06d82eebf046340137b272    93
## 10 7ed6d84e2e6879245e55447aee39c328    85
## # ... with 5,585 more rows
```




```
## # A tibble: 6 x 4
## # Groups:   RescuerID [6]
##   RescuerID                         Species AvgSpeed Count
##   <fct>                             <chr>      <dbl> <int>
## 1 01c59e6ccc820114cdf777c58904a65c Cat            4     1
## 2 020dfd017598e0eb001bfa8b283cd125 Cat            4     1
## 3 021e427ea69bef958fb6222cb3eb0d4a Cat            4     1
## 4 05fabcd5da274c0b654a575430d6a14b Dog            4     1
## 5 08f71e0e4b2f0d37066898f3c3b0a6a5 Cat            4     1
## 6 1080858a732c0bfc16a48175c50e3b5b Dog            4     1
```




It is interesting to note that there are some rescuers who are listed in more than 10 profiles and have faster average adoption speed of the pets they rescued.

### 2.4.18 Photo and video counts

Profile listings can have photos and videos of the pets. A quick check of the data set indicate that the columns PhotoAmt and VideoAmt do not contain invalid values.

```
nrow(df_pets %>% filter(VideoAmt <0))
```

```
## [1] 0
```

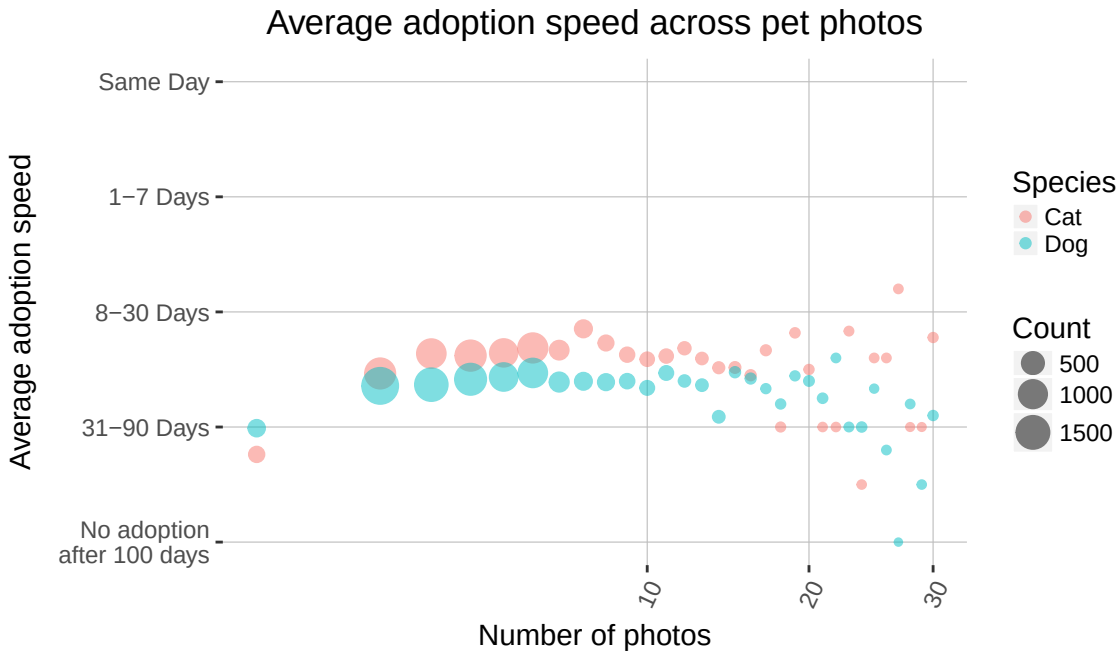
```
nrow(df_pets %>% filter(PhotoAmt <0))
```

```
## [1] 0
```

```
## # A tibble: 31 x 3
##    PhotoAmt Count Average
##       <dbl> <int>   <dbl>
##  1        0   341  0.0227
##  2        1  3075  0.205
##  3        2  2518  0.168
##  4        3  2511  0.167
##  5        4  1881  0.125
##  6        5  2147  0.143
##  7        6   621  0.0414
##  8        7   432  0.0288
##  9        8   314  0.0209
## 10        9   231  0.0154
```

```
## # ... with 21 more rows
```

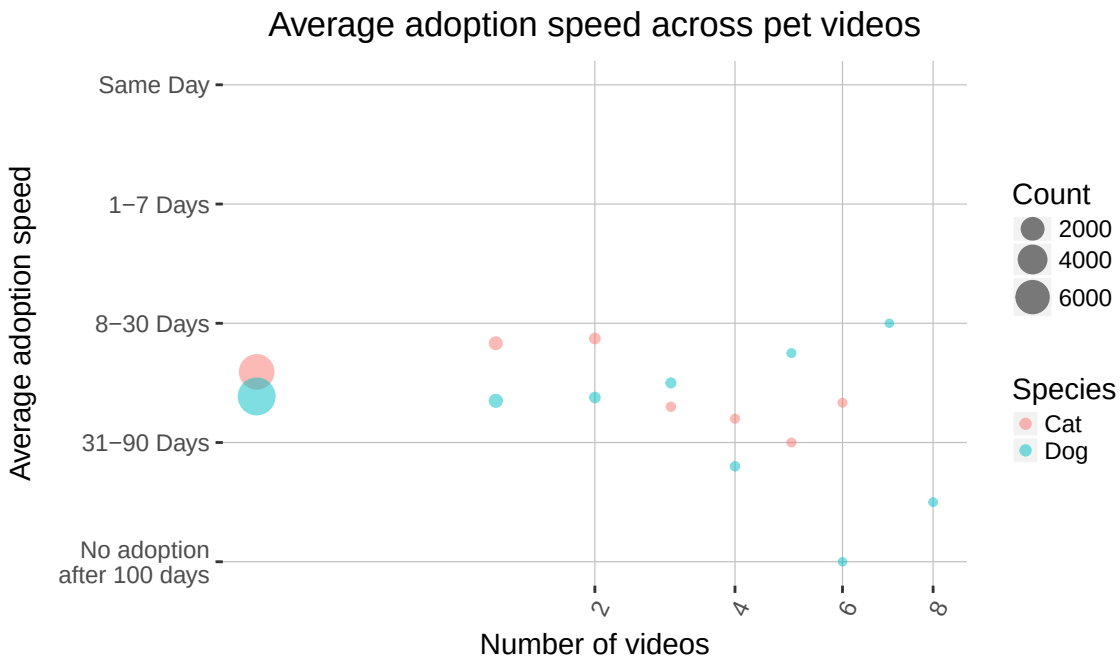
```
## # A tibble: 62 x 4
## # Groups:   PhotoAmt [31]
##    PhotoAmt Species AvgSpeed Count
##       <dbl> <chr>      <dbl> <int>
##  1       27 Cat         2.2      5
##  2        7 Cat         1.85   218
##  3       23 Cat         1.83     6
##  4       19 Cat         1.82    11
##  5       30 Cat         1.78     9
##  6        8 Cat         1.73   137
##  7        5 Cat         1.69  1106
##  8       12 Cat         1.68    57
##  9        6 Cat         1.67   301
## 10       17 Cat         1.67    18
## # ... with 52 more rows
```




```
## # A tibble: 9 x 3
##   VideoAmt Count   Average
##      <dbl> <int>     <dbl>
## 1        0 14419 0.962
## 2        1   417 0.0278
## 3        2    92 0.00614
## 4        3    36 0.00240
## 5        4    15 0.00100
## 6        5     7 0.000467
## 7        6     4 0.000267
```

```
## 8        7     1 0.0000667
## 9        8     2 0.000133
```

```
## # A tibble: 16 x 4
## # Groups:   VideoAmt [9]
##    VideoAmt Species AvgSpeed Count
##       <dbl> <chr>      <dbl> <int>
##  1        7 Dog          2        1
##  2        2 Cat          1.87    47
##  3        1 Cat          1.83   197
##  4        5 Dog          1.75     4
##  5        0 Cat          1.59  6596
##  6        3 Dog          1.5     26
##  7        0 Dog          1.39  7823
##  8        2 Dog          1.38    45
##  9        1 Dog          1.35   220
## 10        6 Cat          1.33     3
## 11        3 Cat          1.3     10
## 12        4 Cat          1.2      5
## 13        5 Cat          1        3
## 14        4 Dog          0.8     10
## 15        8 Dog          0.5      2
## 16        6 Dog          0        1
```




Pets listed with no photo or at all show slower adoption speed in general. Likewise for pets with no video or at all.

### 2.5 Preparing data sets

Some information that are deemed redundant is dropped from the *df_pets* data frame and predictors columns are converted into factors accordingly.

The *df_pets* data frame is validated again to ensure there is no NAs in the data set.

```
##           Type            Age         Breed1         Breed2         Gender
##              0              0              0              0              0
##         Color1         Color2         Color3   MaturitySize     FurLength
##              0              0              0              0              0
##     Vaccinated       Dewormed     Sterilized         Health       Quantity
##              0              0              0              0              0
##            Fee          State       VideoAmt       PhotoAmt Desc_WordCount
##              0              0              0              0              0
##  AdoptionSpeed
##              0
```

### 2.6 Modelling approach

Three models based on caret package's random forest algorithm will be build in this study. The first model will be trained with default settings and search for the best mtry value using all 20 predictors in the original data set with imbalanced classes. Accuracy is used to select the optimal model using the largest value.

The second model will be similar to the first model, with the exception that it will be trained using a data set with balanced classes. This is will be described in detail in the next section.

For the third model, the mtry value will be fixed to the mtry value that produces the best accuracy from the previous model and explore different ntree values.

The data set is split with a 9:1 ratio - into *df_train* that is to be used for training and testing; and *df_validation* for validation.

All the experimental models in this study are build with the aim to optimize each of the model for the best accuracy using *df_train* for training and testing, and validated with *df_validation*. The purpose for validation to see if the models' results remain consistent when valdating it against a new data set (i.e. *df_validation*) that it has never seen during the training and testing stage.

```
## ratio of df_train : df_validation is about  9 : 1
```

The importance of features can be estimated from data used during the model building. Most decision tree algorithms like the random forest have built-in mechanisms to report on variable importance. In this study, varImp is used to show and estimate the variable importance to see if that is close the insights gained during the data exploration and visualization in previous section.

## 3 Results

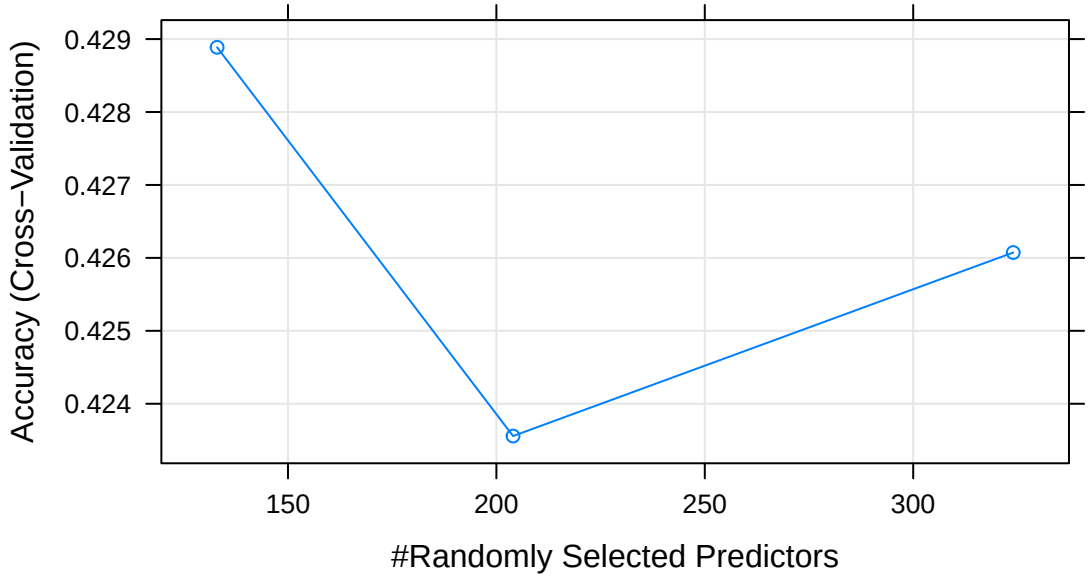
This section describes the results and findings from experimenting with various prediction models.

### 3.1 Model 1: Random forest with 10-fold cross validation method

The first model is built using caret package's random forest algorithm with all 20 predictors from the *df_train* data set to predict AdoptionSpeed class (pet adoption speed) that falls into one of the 5 classes, using 10-fold cross-validation sampling method.

Results from training this first model shows an accuracy of 0.4288873 with mtry = 133 having the highest accuracy.

```
## Random Forest
##
## 13507 samples
##    20 predictor
##     5 classes: '0', '1', '2', '3', '4'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12156, 12156, 12158, 12156, 12156, 12156, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   133   0.4288873  0.2363278
##   204   0.4235571  0.2298895
##   324   0.4260733  0.2334305
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 133.
```




```
##   mtry  Accuracy     Kappa  AccuracySD    KappaSD
## 1  133 0.4288873 0.2363278 0.011061954 0.01431818
```

```
## 2  204 0.4235571 0.2298895 0.009698654 0.01237099
## 3  324 0.4260733 0.2334305 0.014165778 0.01850050
```

The output of variable importance shows the order of importance of variables very much corresponds to what is observed in the previous section of data exploration and visualization with predictors like Breed1, Age, PhotoAmt, Desc_WordCount, Type and Sterilized being ranked as important ones.

```
## rf variable importance
##
##   variables are sorted by maximum importance across the classes
##   only 20 most important variables shown (out of 356)
##
##                     0     1      2     3     4
## Breed1307      100.00 44.20 42.644 37.37 36.42
## Age             87.57 61.54 76.490 79.50 38.23
## PhotoAmt        69.62 66.71 38.155 38.22 33.21
## Desc_WordCount  54.83 65.45 51.673 36.57 31.88
## Type2           52.88 32.19 27.880  0.00 30.77
## Sterilized2     50.57 43.02 52.091 39.17 29.27
## Vaccinated2     32.11 35.86 27.191 45.26 25.76
## Breed2307       43.14 40.64 11.610  4.64 15.33
## Dewormed3       34.39 42.23  8.124 28.88 18.23
## State41326      42.16 38.23 27.618 29.12 26.79
## Quantity        42.14 37.32 28.636 29.92 23.70
## Sterilized3     39.42 20.96 23.466 27.22 25.73
## Fee             33.11 33.44 37.825 28.18 19.06
## MaturitySize2   35.27 29.64 36.451 36.99 31.56
## Vaccinated3     30.56 35.76 12.632 29.24 17.15
## Breed226        15.61 13.30 34.805 14.24 13.31
## State41327      29.26 34.61 32.438 20.54 16.26
## FurLength2      27.61 34.38 32.637 28.31 19.87
## Dewormed2       30.09 34.23 21.847 34.32 23.99
## Breed2247       14.46 14.01 13.672 16.38 34.25
```

This first prediction model is also tested on the *df_validation* data set where the training set (*df_train*) has never been exposed to.

```
## prediction_1
##   0   1   2   3   4
## 537 236 430 279   4
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2   3   4
##          0 267  90 105  64  11
##          1  36  89  70  35   6
##          2  78  99 146  99   8
##          3  36  44  80 108  11
##          4   0   0   0   0   4
```

```
##
## Overall Statistics
##
##                Accuracy : 0.4132
##                  95% CI : (0.388, 0.4387)
##     No Information Rate : 0.2806
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2148
##  Mcnemar's Test P-Value : 7.936e-13
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.6403  0.27640  0.36409  0.35294 0.100000
## Specificity            0.7474  0.87371  0.73825  0.85508 1.000000
## Pos Pred Value         0.4972  0.37712  0.33953  0.38710 1.000000
## Neg Pred Value         0.8419  0.81360  0.75852  0.83596 0.975709
## Prevalence             0.2806  0.21669  0.26985  0.20592 0.026918
## Detection Rate         0.1797  0.05989  0.09825  0.07268 0.002692
## Detection Prevalence   0.3614  0.15882  0.28937  0.18775 0.002692
## Balanced Accuracy      0.6939  0.57505  0.55117  0.60401 0.550000
```

Results from validating this first model against the *df_validation* data set shows an accuracy of 0.4131898 that is close to the training set. This re-assures the consistency of the model.

| model | accuracy | kappa |
|---|---|---|
| model 1: random forest, cv=10, mtry=133 | 0.4288873 | 0.2363278 |

### 3.2 Model 2: Dealing with Imbalanced Classes

In most real-world classification problems, sample data sets may have some level of class imbalance, which is when each class does not make up an equal portions in the data set. In this case, it is crucial to properly adjust metrics and methods accordingly.

As mentioned in previous section, one of the classes (AdoptionSpeed of 4, which indicates the pet is adopted on the same day it is listed) is not evenly distributed like the other classes.

```
## # A tibble: 5 x 3
##   AdoptionSpeed     n  ratio
##   <fct>         <int>  <dbl>
## 1 0              4197 0.280
## 2 1              3259 0.217
## 3 2              4037 0.269
## 4 3              3090 0.206
## 5 4               410 0.0273
```

This AdoptionSpeed class of value 4 is only about 2.7% of the sample size.

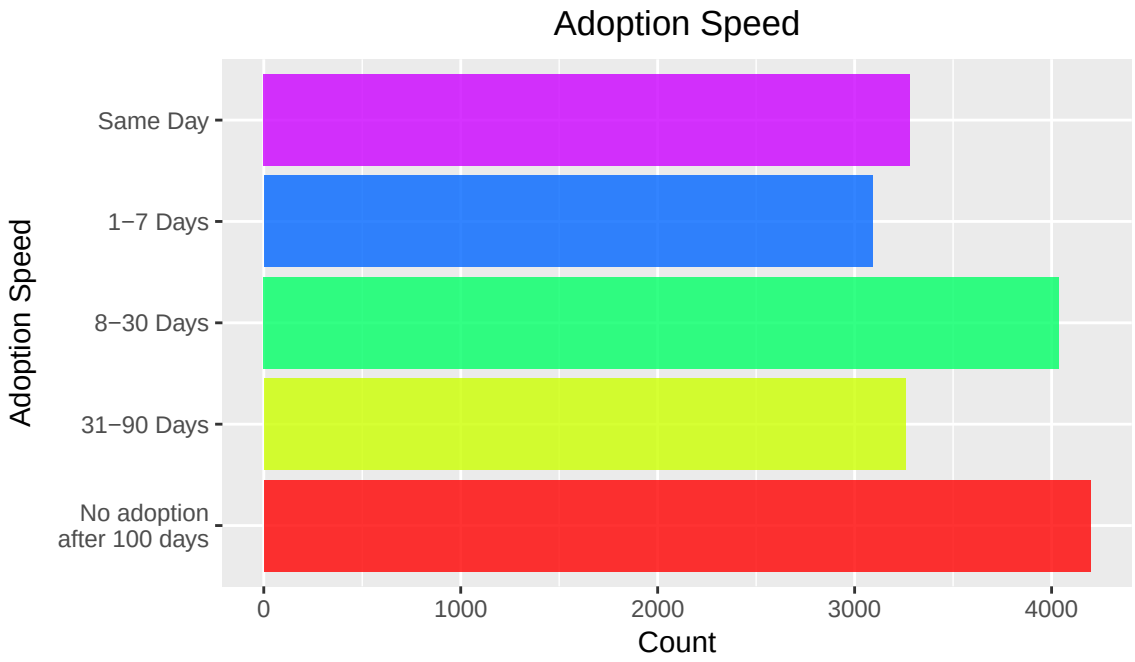
A simple way to fix imbalanced data set is simply to balance them, either by oversampling instances of the minority class or undersampling instances of the majority class. In this project, the AdoptionSpeed class of 4 will be oversampled to balance it out with the rest of the classes. The *df_pets* data frame is reconstructed (restored from a backup data frame *df_pets_backup*) and the AdoptionSpeed class of 4 being replicated 7 times to match the distribution of the other classes of AdoptionSpeed.

It is worth noting that, this simple method has its flaws. Oversampling a minority class can lead to model overfitting, since it will introduce duplicate instances, drawing from a pool of instances that is already small. Likewise, undersampling would also introduce bias as samples with good value may be discarded.

There are other methods to approach imbalanced data sets such as Synthetic Minority Oversampling (SMOTE) technique; or using Kappa coefficient as metric of measurement instead of accuracy. These sampling method and metric will not be in the scope of this project.

```
## Samples with AdoptionSpeed==4 is 410
```

```
## # A tibble: 5 x 3
##   AdoptionSpeed Count  Rate
##   <chr>         <int> <dbl>
## 1 4              3280 0.184
## 2 3              3090 0.173
## 3 2              4037 0.226
## 4 1              3259 0.182
## 5 0              4197 0.235
```

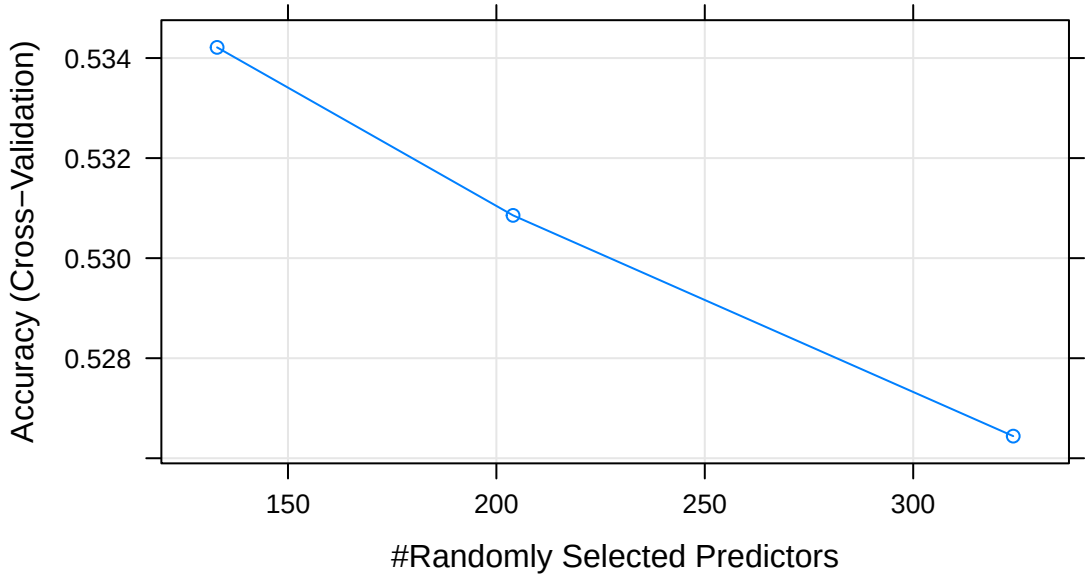



The *df_train* and *df_validation* data sets are re-split like previous.

```
## ratio of df_train : df_validation is about  9 : 1
```

This second model that is trained with a more evenly distributed set of prediction classes (AdoptionSpeed) in the data set has accuracy of 0.5342133 with mtry = 133 having the best accuracy. The experiment results show that having a somewhat balanced portions of classes does improve the accuracy of this model when compared to the previous model which was trained with imbalanced portions of classes. The value of Kappa coefficient also improved greatly in this second model, compared to the first model; although this model is not tuned to optimize on Kappa coefficient.

```
## Random Forest
##
## 16089 samples
##    20 predictor
##     5 classes: '0', '1', '2', '3', '4'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 14480, 14480, 14482, 14480, 14479, 14480, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   133   0.5342133  0.4127922
##   204   0.5308551  0.4087871
##   324   0.5264420  0.4035606
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 133.
```




```
##   mtry  Accuracy     Kappa  AccuracySD    KappaSD
## 1  133 0.5342133 0.4127922 0.008439877 0.01048796
## 2  204 0.5308551 0.4087871 0.010929429 0.01370242
## 3  324 0.5264420 0.4035606 0.010524272 0.01315791
```

The output of variable importance shows the order of importance of variables very much corresponds to what is observed in the first model and previous section of data exploration and visualization with predictors like Age, PhotoAmt, Desc_WordCount and Breed1 being ranked as top important ones.

```
## rf variable importance
##
##   variables are sorted by maximum importance across the classes
##   only 20 most important variables shown (out of 356)
##
##                      0      1      2      3      4
## Desc_WordCount 18.914 24.866 19.635 12.190 100.00
## Age            33.263 24.440 30.022 28.572  87.91
## PhotoAmt       21.861 24.857 14.466 12.053  73.96
## Breed1307      40.036 15.506 13.789 10.879  41.84
## Gender2         9.344 10.056  4.952  4.835  41.29
## Fee            12.170 11.343 14.387 10.260  39.81
## Quantity       13.994 14.209  9.497  8.905  39.70
## Color27         4.426  7.803  5.916  4.793  37.51
## FurLength2      8.056 11.746  9.954  9.288  35.70
## State41401      6.852  8.828  7.438  9.594  35.64
## Color15         6.644  3.837  3.949  7.644  32.20
## Color37         5.992  5.823  6.143  5.565  31.94
## Color22         3.189  8.073  8.050  6.782  31.91
## State41326     13.535 12.956 10.301  9.039  31.88
## Color12         5.936  9.224  7.771  5.395  31.74
## Breed2247       4.773  2.519  7.070  5.522  31.54
## VideoAmt        7.875  3.964  3.506  5.182  30.38
## FurLength3     12.364  8.533  7.599  7.962  29.65
## Color16         2.922  5.827  6.454  3.956  29.58
## Color35         2.528  4.700  2.635  5.108  29.46
```

This prediction model is also tested on the *df_validation* data set where the training set (*df_train*) has never seen.

```
## prediction_2
##   0   1   2   3   4
## 527 223 415 252 357
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2   3   4
##          0 266  85 108  68   0
##          1  31  89  70  33   0
##          2  80  97 141  97   0
##          3  36  44  74  98   0
##          4   4   7   8  10 328
##
## Overall Statistics
```

```
##
##                Accuracy : 0.5197
##                  95% CI : (0.4962, 0.5432)
##     No Information Rate : 0.2351
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3942
##  Mcnemar's Test P-Value : 1.785e-12
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.6379  0.27640  0.35162  0.32026   1.0000
## Specificity            0.8077  0.90771  0.80044  0.89510   0.9799
## Pos Pred Value         0.5047  0.39910  0.33976  0.38889   0.9188
## Neg Pred Value         0.8789  0.84977  0.80868  0.86334   1.0000
## Prevalence             0.2351  0.18151  0.22604  0.17249   0.1849
## Detection Rate         0.1499  0.05017  0.07948  0.05524   0.1849
## Detection Prevalence   0.2971  0.12570  0.23393  0.14205   0.2012
## Balanced Accuracy      0.7228  0.59206  0.57603  0.60768   0.9900
```

Validation result of this second model against the *df_validation* data set has an accuracy of 0.5197294 that is close to the training data set. This re-assures the consistency of the model.

The sensitivity for class AdoptionSpeed 4 has also greatly improved with this model that is trained with data set that have a somewhat balanced portion of classes.

| model | accuracy | kappa |
|---|---|---|
| model 1: random forest, cv=10, mtry=133 | 0.4288873 | 0.2363278 |
| model 2: re-train model 1 using data set with balanced portion of classes | 0.5342133 | 0.4127922 |

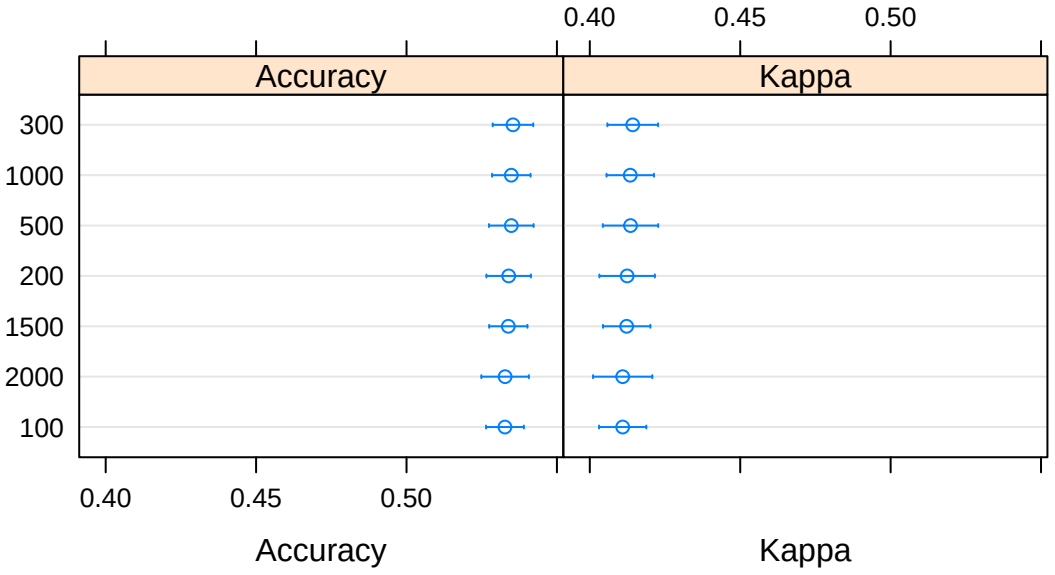
### 3.3 Model 3: Experimenting with ntree parameter

The third model is build based on the previous model and fixing the parameter mtry = 133. This third model will study how the different ntree values of 100, 200, 300, 500 (default ntree value), 1000, 1500 and 2000 would affect the accuracy.

Results show that ntree value of 300 trees gives the best accuracy.

```
##
## Call:
## summary.resamples(object = model_3_ntree)
##
## Models: 300, 500, 1000, 1500, 100, 200, 2000
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
```

```
## 300  0.5233064 0.5300151 0.5340378 0.5353935 0.5399899 0.5561763    0
## 500  0.5220634 0.5298597 0.5321733 0.5348333 0.5375288 0.5580385    0
## 1000 0.5276569 0.5292107 0.5320274 0.5348338 0.5352809 0.5567970    0
## 1500 0.5251709 0.5288999 0.5315512 0.5338398 0.5359969 0.5549348    0
## 100  0.5208204 0.5250243 0.5318619 0.5327217 0.5399188 0.5468653    0
## 200  0.5220634 0.5280675 0.5310752 0.5339629 0.5358195 0.5561763    0
## 2000 0.5233064 0.5283606 0.5288999 0.5327823 0.5356510 0.5611421    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## 300  0.3992443 0.4072071 0.4129242 0.4142681 0.4202026 0.4401745    0
## 500  0.3977244 0.4068358 0.4106372 0.4135296 0.4167682 0.4422315    0
## 1000 0.4047187 0.4064270 0.4099858 0.4134471 0.4138941 0.4405075    0
## 1500 0.4015407 0.4063739 0.4093085 0.4122511 0.4147873 0.4383405    0
## 100  0.3959981 0.4014178 0.4101639 0.4109266 0.4200238 0.4282435    0
## 200  0.3978188 0.4052473 0.4090382 0.4124119 0.4146435 0.4400828    0
## 2000 0.3987699 0.4056567 0.4062642 0.4109247 0.4143066 0.4462932    0
```




```
## $`300`
## Random Forest
##
## 16089 samples
##    20 predictor
##     5 classes: '0', '1', '2', '3', '4'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 14480, 14480, 14482, 14480, 14479, 14480, ...
## Resampling results:
```

```
##
##   Accuracy   Kappa
##   0.5353935  0.4142681
##
## Tuning parameter 'mtry' was held constant at a value of 133
```

```
##   mtry  Accuracy     Kappa  AccuracySD    KappaSD
## 1  133 0.5353935 0.4142681 0.009423275 0.01177208
```

The result shows that setting ntree value to 300 produces the best accuracy among the ntree values explored in this case. Kappa coefficient is also the highest for ntree=300 although the model is not optimized for Kappa coefficient value in this study.

The output of variable importance shows the order of importance of variables very much corresponds to what is observed in the first two models and also previous section of data exploration and visualization with predictors like Age, PhotoAmt, Quantity and Desc_WordCount now being ranked as important ones.

```
## rf variable importance
##
##   only 20 most important variables shown (out of 356)
##
##                Overall
## Desc_WordCount 100.000
## Age             54.738
## PhotoAmt        54.500
## Quantity        18.274
## Fee             17.246
## Gender2         16.318
## FurLength2      14.614
## Color12         13.397
## State41326      13.396
## Color27         13.232
## Dewormed2       11.777
## State41401      11.185
## MaturitySize2   10.982
## Color37         10.974
## Color22         10.824
## Vaccinated2     10.667
## Breed1307        8.616
## Sterilized2      8.346
## Color25          6.611
## Color15          6.609
```

This third prediction model is also tested on the *df_validation* data set where the training set (*df_train*) has never seen. The results shows a close accuracy range and this re-assures that the model is consistent.

```
prediction_3 <- predict(model_3, df_validation)
table(prediction_3)
```

```
## prediction_3
##   0   1   2   3   4
## 526 221 411 260 356
```

```
df_prediction_3 <- as.data.frame(prediction_3)
cm_3 <- confusionMatrix(df_prediction_3$X300, df_validation$AdoptionSpeed)
print(cm_3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2   3   4
##          0 268  85 107  66   0
##          1  34  84  67  36   0
##          2  74 101 141  95   0
##          3  36  45  79 100   0
##          4   5   7   7   9 328
##
## Overall Statistics
##
##                Accuracy : 0.5192
##                  95% CI : (0.4956, 0.5426)
##     No Information Rate : 0.2351
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3936
##  Mcnemar's Test P-Value : 7.282e-12
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.6427  0.26087  0.35162  0.32680   1.0000
## Specificity            0.8099  0.90565  0.80335  0.89101   0.9806
## Pos Pred Value         0.5095  0.38009  0.34307  0.38462   0.9213
## Neg Pred Value         0.8806  0.84675  0.80924  0.86394   1.0000
## Prevalence             0.2351  0.18151  0.22604  0.17249   0.1849
## Detection Rate         0.1511  0.04735  0.07948  0.05637   0.1849
## Detection Prevalence   0.2965  0.12458  0.23168  0.14656   0.2007
## Balanced Accuracy      0.7263  0.58326  0.57749  0.60890   0.9903
```

Validation of the third model against the *df_validation* data set shows accuracy value of 0.5197294.

| model | accuracy | kappa |
|---|---|---|
| model 1: random forest, cv=10, mtry=133 | 0.4288873 | 0.2363278 |
| model 2: re-train model 1 using data set with balanced portion of classes | 0.5342133 | 0.4127922 |
| model 3: tune mtry=133 and ntree=300 | 0.5353935 | 0.4142681 |

The third model shows that ntree parameter of 300 trees has the best accuracy and Kappa coefficient

among the ntree values explored in this case. This experiement with the third model explores different ntree values and result shows that having the ntree parameter set to 300 can improve the accuracy; although it is marginal. In this case, since such a small ntree value can help to improve the accuracy of the prediction model; this finding is helpful for studies carried out with limited computing resource and also helps reduce time in building the model.

In some cases, tuning the number of trees may be unnecessary; instead, simply set the number of trees to a large, computationally feasible number would suffice.

# 4 Conclusion

This project shows that a popular machine learning algorithm can help in predicting 5 different classes of pet adoption speed based on metadata associated with their online profile listing. Although the accuracy of the prediction model is not impressive, slightly above 0.5, it is still better than random guess which would have accuracy that is between 0.2 and 0.25.

As mentioned in previous sections and demonstrated in the second model, the sample sizes of the 5 adoption speed classes are not evenly distributed. More work could be done to optimize the prediction model based on metrics that are not sensitive to uneven distribution of prediction classes, such as the Kappa coefficient.

More work could also be done to explore how differnt prediction models will behave when they are trained with data set that is balanced using different methods such as SMOTE to balance the classes in the data set.

This study uses all 20 predictors in the data set to train and experiment with different prediction models with the exception of few columns that considered redundant and are dropped, such as PetID, RescuerID and Name. More work could be done in studying how different number of predictors can help improve the performance of building the prediction models whilst keeping the accuracy of the models at an acceptable level. This can help to reduce the amount of time and computing resource required to build the prediction models.