Children-Friendliness of Adoptable Dogs

Authors

- **Yu-Sian Lin**
  · ⦿ yslin0114
  Department of Civil Engineering, University of Illinois

- **Yueh-Ti Lee**
  · ⦿ Yueh-Ti
  Department of Civil Engineering, University of Illinois

- **Minjiang Zhu**
  · ⦿ Minjiang-Zhu
  Department of Something, University of Illinois

- **Chengyou Yue**
  · ⦿ TensorYue
  Department of Civil Engineering, University of Illinois

Description:

Source: Kaggle/ Adoptable Dogs in the US

Data Format: CSV File

Content: The dataset contains information on over 3,000 adoptable dogs listed on PetFinder.com regarding to their origins, as well as the state they are currently listed for adoption in.

Attributes:

contact_city: The city where the animal is located. (String)

contact_state: The state where the animal is located. (String)

description: A description of the animal. (String)

url: The URL of the animal's profile on PetFinder. (String)

type.x: The type of animal. (String)

species: The species of the animal. (String)

breed_primary: The primary breed of the animal. (String)

breed_secondary: The secondary breed of the animal. (String)

breed_mixed: Whether the animal is a mixed breed. (String)

breed_unknown: Whether the animal's breed is unknown. (String)

color_primary: The primary color of the animal. (String)

color_secondary: The secondary color of the animal. (String)

color_tertiary: The tertiary color of the animal. (String)

age: The age of the animal. (String)

sex: The animal's sex. (String)

size: The size of the animal. (String)

coat: The type of coat the animal has. (String)

fixed: Whether the animal is spayed or neutered. (String)

house_trained: Whether the animal is house trained. (String)

declawed: Whether the animal is declawed. (String)

special_needs: Whether the animal has any special needs. (String)

shots_current: Whether the animal is up to date on shots. (String)

env_children: Whether the animal is good with children. (String)

Proposal:

We plan to develop a model that can predict the extent of children-friendliness of adoptable dogs based on the dog's origin information, such as the dog's breed, color, age and so on, so that families with children can take this model for reference when planning to adopt a dog. We will start by finding the correlation between the dog's features and env_children in the dataset. We will then evaluate the indices to obtain the extent of the children-friendliness of a dog.

Exploratory Data Analysis

A persuasive prediction model needs to be trained on the data with enough scale. Thus, before we move to the model part, we need to figure out the distribution of data itself.

First, we want to know the amount of data of each breed. Notice that many mixed breed dogs miss their secondary breed, so instead of considering secondary breeds, we only focus on their primary breeds. The sorted amount of each primary breed is shown in Figure 1.
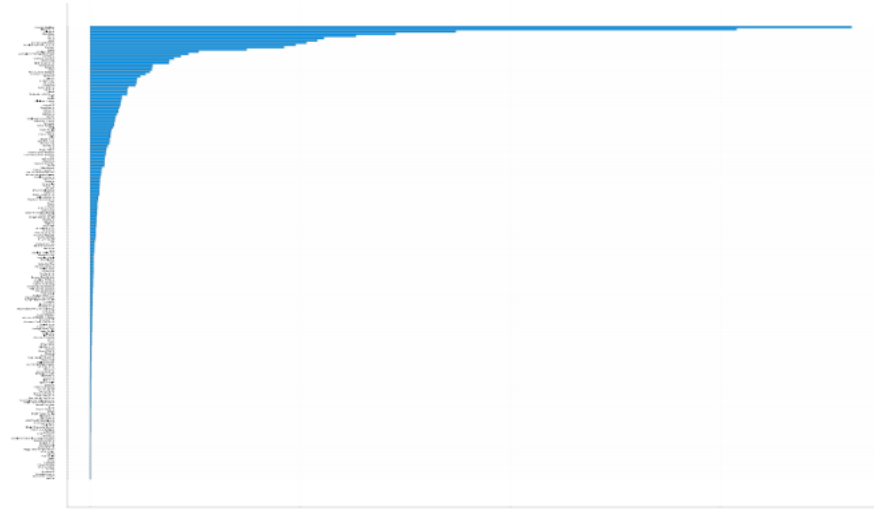

Figure 1. Amount of each Primary Breed

There are 203 primary breeds, however, most of them do not have enough data to support for a meaningful model. We choose those amount that are larger than 1000, which is shown in Table 1.

Primary Breed	Amount
Labrador Retriever	3625
Pit Bull Terrier	3077
Chihuahua	1740
Mixed Breed	1454
Terrier	1264
Hound	1112
Boxer	1080
German shepherd	1029

Table 1. Amount of each Primary Breed (Only shows those amount that are larger than 1000)

After sorting the amount, we want to have a brief knowledge of each breed with their children friendliness tend. Here, we made a group bar plot that shows the fraction of children friendliness with respect to their primary breed.
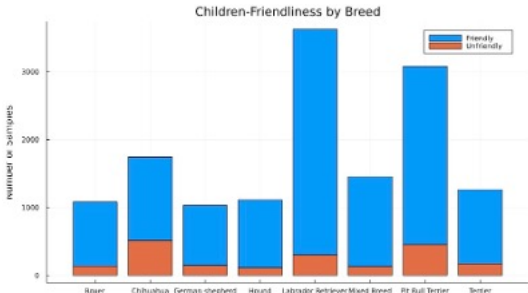

Figure 2. Children-Friendliness by Breed

We also want to know how efficient the actions (fixing or house training) would be to make each breed to be friendly with children. Thus, we collected those dogs with and without fixed or trained to make another grouped bar plot. We only show data without any actions, the actions process would be the same as this.
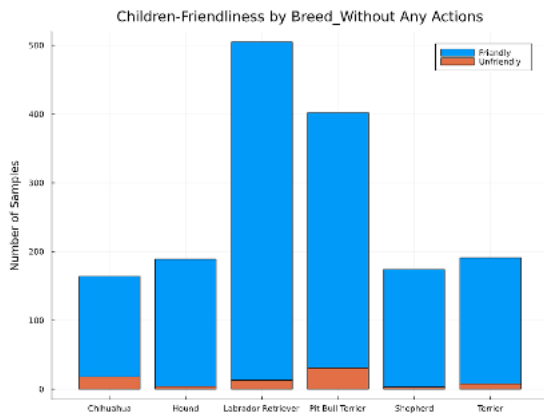
Figure 3. Children-Friendliness by Breed without actions

Finally, considering some breeds might be larger on size, or the amount of data in particular age, particular sex and particular size might be very small. Thus, we need to know the distribution of our data in each combination. Here we only show the distribution of "Labrador Retriever", the process is the same in other breeds.
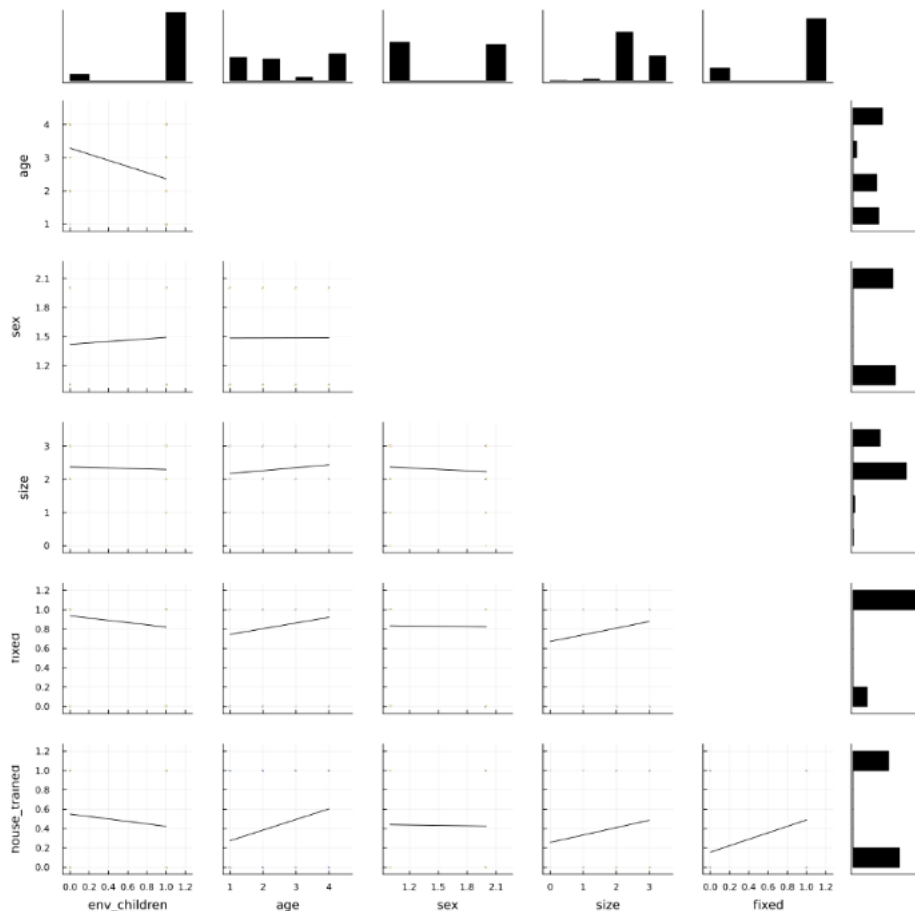

Figure 4. The distribution of "Labrador Retriever" in each combination

We could know that the size of "Labrador Retriever" usually "Large" or "Extra Large", so we could only focus on these sizes when establishing the prediction model of this breed.

We could also realize that the fixed "Labrador Retriever" is much more than unfixed (around 20%). But considering the scale of our dataset (3625), we could still have 725 data. We say that this is enough to establish a persuasive prediction model.

Predictive Modeling

Here we do some basic analysis on linear model, to discuss the accuracy of the linear model. Then establish two nonlinear models based on two types of learning method for this problem.

[Linear Model] First, we think about the linear regression model: Xw + b = P We have 5 categories in this problem, every observation is the combination of each category, and each category would have 2 branches. Thus, the total number of the combinations would be 2^5. For a generalized type of problem with m categories, the total number would be $\prod_{i=1}^{m} n_i$, where n_i is the number of branches in each category. - Here X is a m×m matrix and w is a vector with n component. It is naturally to ask the following question: is this model perform a proper approximation of the real problem? To answer

this question, we extend our independent variables to $N = \prod_{i=1}^{m} n_i$. In this case, each branch would be independent. We would have a generalized model for the linear regression. $\check{X} \tilde{w} = \check{P}$. - Here $\check{X}$ is a N×N matrix and $\tilde{w}$ is a vector with N components. For the linear regression process, we are forcing $\check{P}$ to be the expectation of our supervised result $\check{P}$=E(y). Thus, we can evaluate our $\check{X}$ in the following procedure.

```
A=zeros(N,N); % Create store matrix
count=1; % Create ID of row for updating
for i_1=1:n_1
    ... % m Loops over each branch
    for i_m=1:n_m
        A(count,i_1+t_1)-1; % t_num is the number of branches of previous categories
        ... % Evaluate
        A(count,i_m+t_m)=1;
        count=count+1; % Update ID of row
    end
    ...
end
rank(A) % Calculate the rank of the matrix
```

We can say that

$$rank(A) = R = 1 + \sum_{i=1}^{m}(n_i - 1)$$

. Thus, for linear regression with Xw+b=P, we should have X is a R×R matrix and w is a vector with R components. However, this is not a proper model for such a problem. In this model, we have N expectations and rank(A)<N. We need to establish a better model to capture the nonlinearity of the problem.
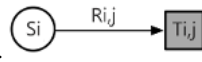
[ Reinforcement Learning - Monte Carlo Method ] Here we introduce a model which could be suitable for the nonlinearity. In Reinforcement Learning, for discretized system, we have the definition of state S, action a, reward R, value of state V and value of action Q. Imagine that we are in a grid world, state $S_i$ is where we stand. We have the list of actions in each state, here we can go [up, down, left, right]. When we take an action in each state, we would get a reward $R_i$ from our chosen action and move to the new state $S_1$. We do this repeatedly until we reach the terminal of this world, end would occur then. After each end, we would generate an episode which stores the states that we have been visited.
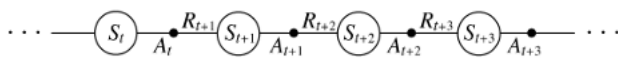


The value of state:

$$V^*(S) = \max_a \left( R(S,a) + \gamma \sum_{S'} V^*(S') \right)$$

$$= \max_a \left( R(S,a) + \gamma \sum_{S'} T(S,a,S') \max_{a'} R(S',a') + \cdots \right)$$

Here T(S,a,S') is the transition probability, the probability of reaching S' while we start from S and take action a. γ is the discounting rate. In this generalized problem, we have N parallel states, and directly related to their own terminal without any actions. $S_i$ is the fixed state, $R_{(i,j)}$ is the reward of $S_i$ at observation j, $T_{(i,j)}$ is the terminal of $S_i$ at observation j. We can set $R_{(i,j)}$=0 and $V(T_{(i,j)})$=1 when it is a target result. Then we would update it incrementally:

$$V(S_i) = \frac{j-1}{j} V(S_i) + \frac{1}{j} V(T_{i,j}) = V(S_i) + \frac{1}{j} [V(T_{i,j}) - V(S_i)]$$

We can also add discounting rate γ (more like learning rate here) into our update function:

$$V(S_i) = V(S_i) + \frac{\gamma}{j} [V(T_{i,j}) - V(S_i)]$$

[ Machine Learning – Neural Network ] We can set the input number of the first layer as

$$rank(A) = R = 1 + \sum_{i=1}^{m}(n_i - 1)$$

as our previous analysis and add any number of layers and nonlinear functions between each layer as we want. The final output number should be 1 based on our problem type. Py-Torch is a useful tool for machine

```
Net(
    (fc1): Linear(in_features=32, out_features=50, bias=True)
    (fc2): Linear(in_features=50, out_features=20, bias=True)
    (fc3): Linear(in_features=20, out_features=1, bias=True)
)
```

learning, here we give a sample structure of neural network.

References:

Adoptable Dogs in the US | Kaggle Reinforcement Learning: An Introduction by Richard S. Sutton CEE498LM by Prof. Alireza

References

Energy Efficiency of buildings in New York