Pet Adoption Prediction

By Jessica Lewis

Problem

Can we predict how long until an animal is adopted based on their location and the characteristics listed on their online profile? Using these predictions and extrapolating the important features, can we fine-tune an animal's online profile to increase their rate of adoption?

Approach

The Data

Because I'm working with a fictional company I don't have "real" data. Instead I used the Petfinder API and downloaded all dogs and cats in Washington state that were flagged as adopted in 2019. I used 2019 because it's the last "normal" year and I decided that my hypothetical scenario does not exist during a complicated worldwide pandemic.

I also wanted to be able to find some stats per city, since the petfinder data comes with some location information. For this I used the United States Census API to grab some basic demographic census data. I filtered the location to cities in Washington state and then used an inner join to exclude animals adopted out of state as well as cities that had no adoption data.

Feature Engineering

The data came in with two date fields: published_at and status_changed_at. These describe when an animal was added to the database and when their status was changed from "available" to "adopted".
Since the goal of this project is to predict how long an animal will have to wait before being adopted I needed a column for that metric. To do this I subtracted 'published_at' from 'status_changed_at' to make a new column called 'duration_as_adoptable'. This is the **dependent variable**.

Data Cleaning and Preprocessing

Firstly, cats and dogs are processed separately. Each step had to be completed for each species. Because of this I ended up writing several functions so that I didn't have to copy/paste so much code. I made one function with the intention of cleaning up all of the training and test data, as well as any future imports so that the data can be run straight through the model. This included pulling dictionaries into their own fields, dropping some unnecessary columns, and removing duplicate rows.
The training and test sets each had some additional work done. I took care of all missing data, and ported over city names from the organization ids in each record. I used the petpy org API to grab city names and then mapped over the census population data. I also trimmed out some outliers and scaled the data, then converted the categorical fields into dummies.

You can see my custom functions here:
https://github.com/JessicaELewis/Pet-Adoptions/blob/main/library/preprocess.py

Feature Selection

I tried three different methods for feature selection and then manually re-saved the dataframe with only those features. The models I used for feature selection are:

- RandomForest
- XGBoost
- f_regression

Because I had a limited amount of fields overall, and I used dummies in feature selection, I trimmed out the specific values that get_dummies had created and that the models selected and ended up using all of the fields that came in with the data. This is a point where I should have paid more attention; going back and adding some more features either by engineering or by bringing in more demographic data could have yielded a more refined set of features.

Model Selection

I tested four models and assessed their performance on the data:

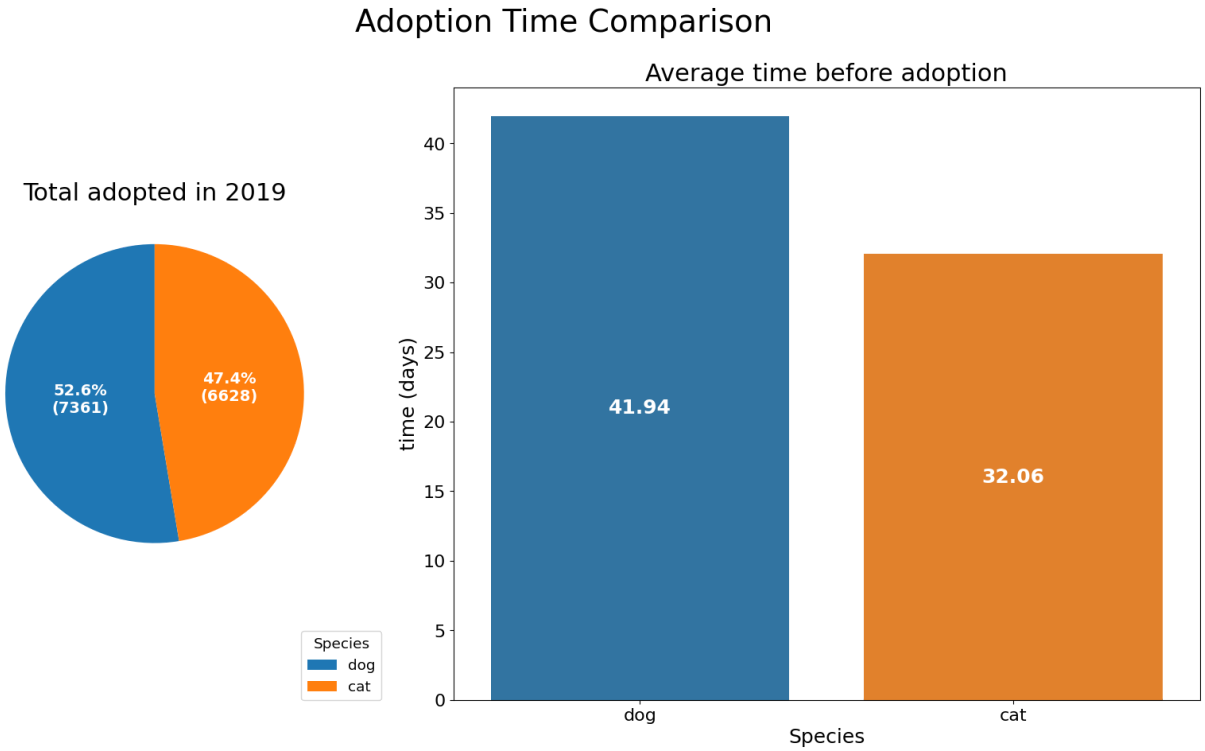
- RandomForestRegressor

- GradientBoostingRegressor
- KNeighborsRegressor
- xgboost

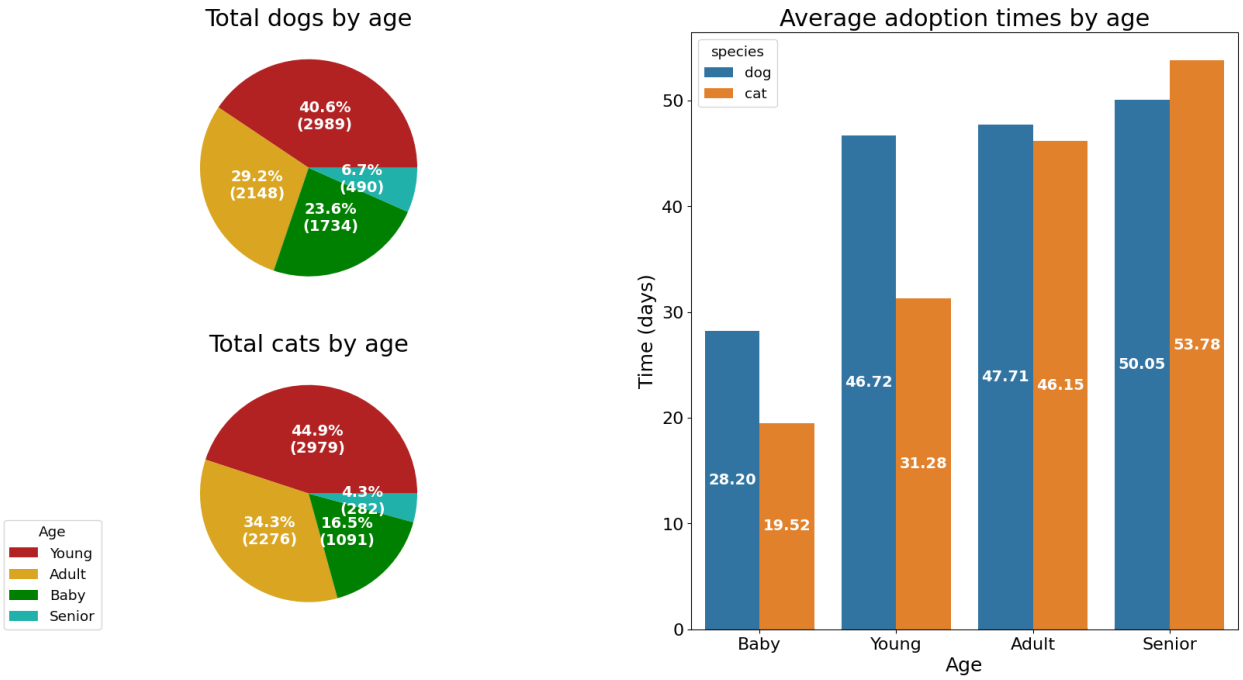
Hyperparameter Tuning

I ended up doing parameter tuning on each of these models for both cats and dogs because none of the models performed very well, and none of the models stood out from the rest. This was the case even after tuning, but overall the performance did increase a little.
By this point I had spent way more time than I had planned on this project, so I ended up using the same hyperparameters for the final models, even though I could have tuned more. I'm not sure how much of a difference tuning more would have had on performance; I think other improvements, like adding more features for selection, would have been more effective.

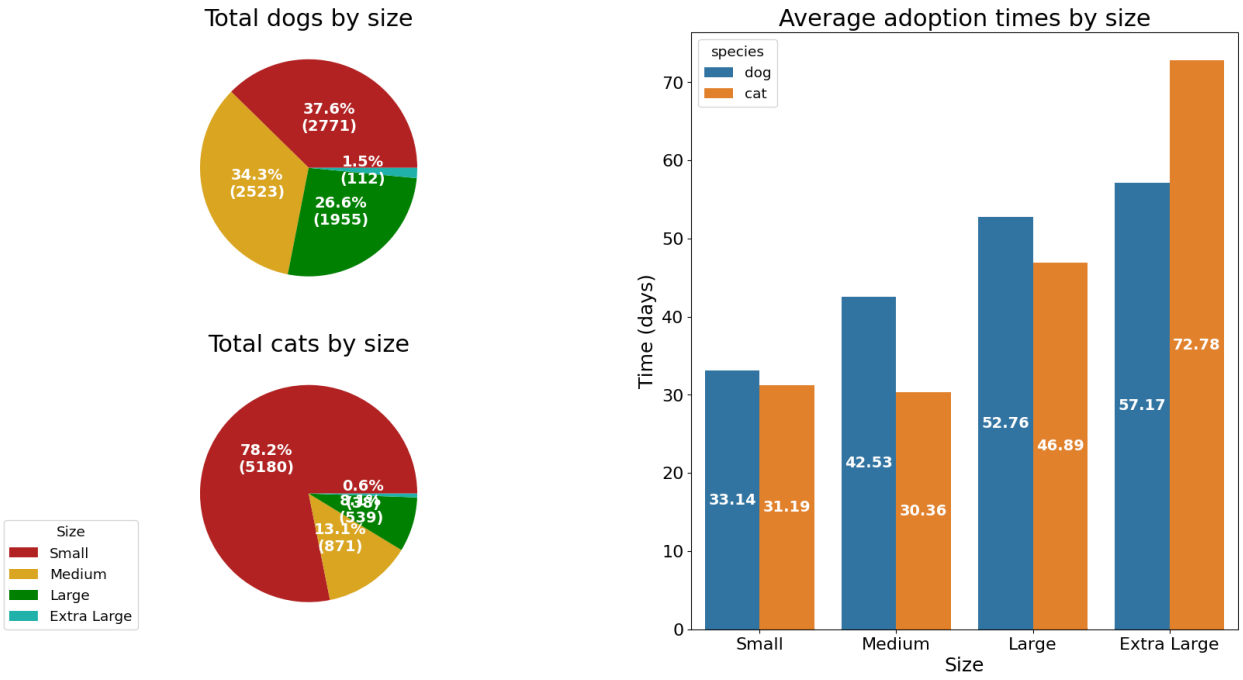
EDA



As you can see on the left, more dogs were adopted in 2019 than cats. Without any context it's hard to say if dogs are preferred or if there were just more dogs available for adoption. Although more dogs were adopted, the average time before adoption was about 30% longer than cats. Again, I can't say for sure if this is because cats were sought out or if this imbalance was because there were simply more dogs to choose from. The latter seems like a reasonable assumption, but I wouldn't make any recommendations based solely on this statistic.

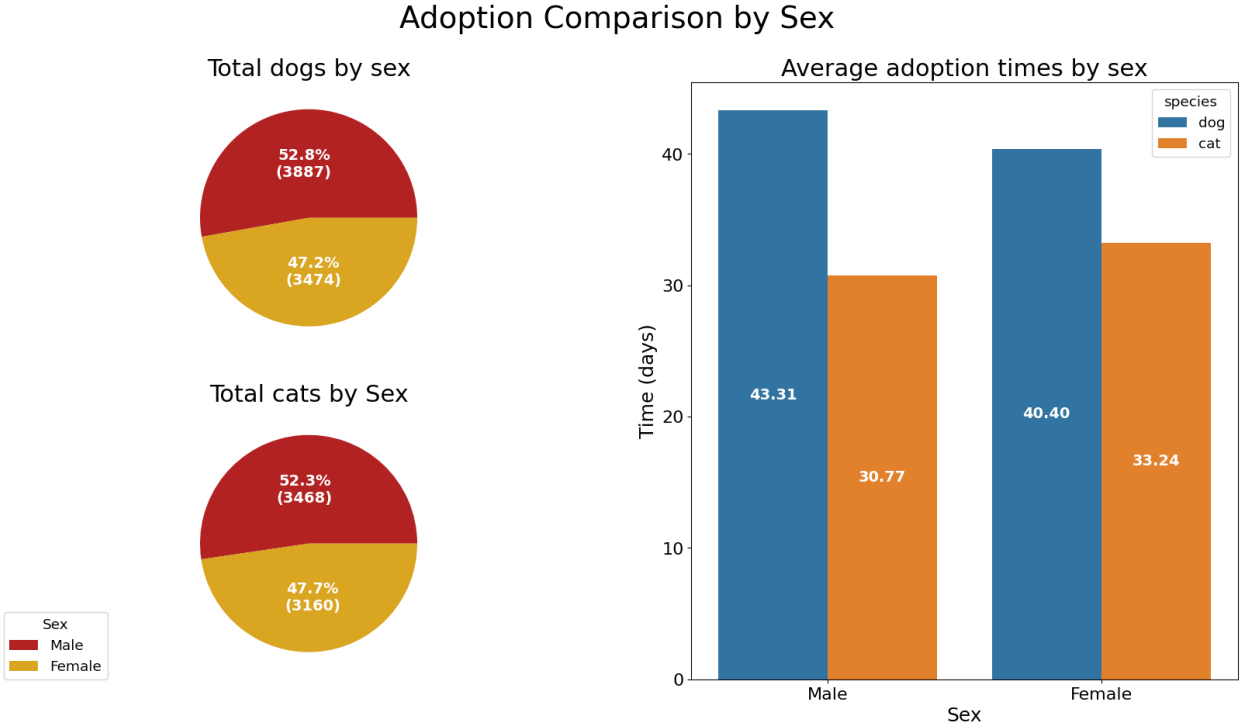


According to our totals on the left, young dogs and cats were the most adopted age range in 2019, followed by adult dogs and cats, then baby and senior. Despite this, the bar chart indicates that baby animals are, predictably, the fastest to be adopted. After that you can see that dogs' adoption rate evens out while cats' adoption rate slows continuously. Finally, it seems that senior cats spend the most time overall in shelters.

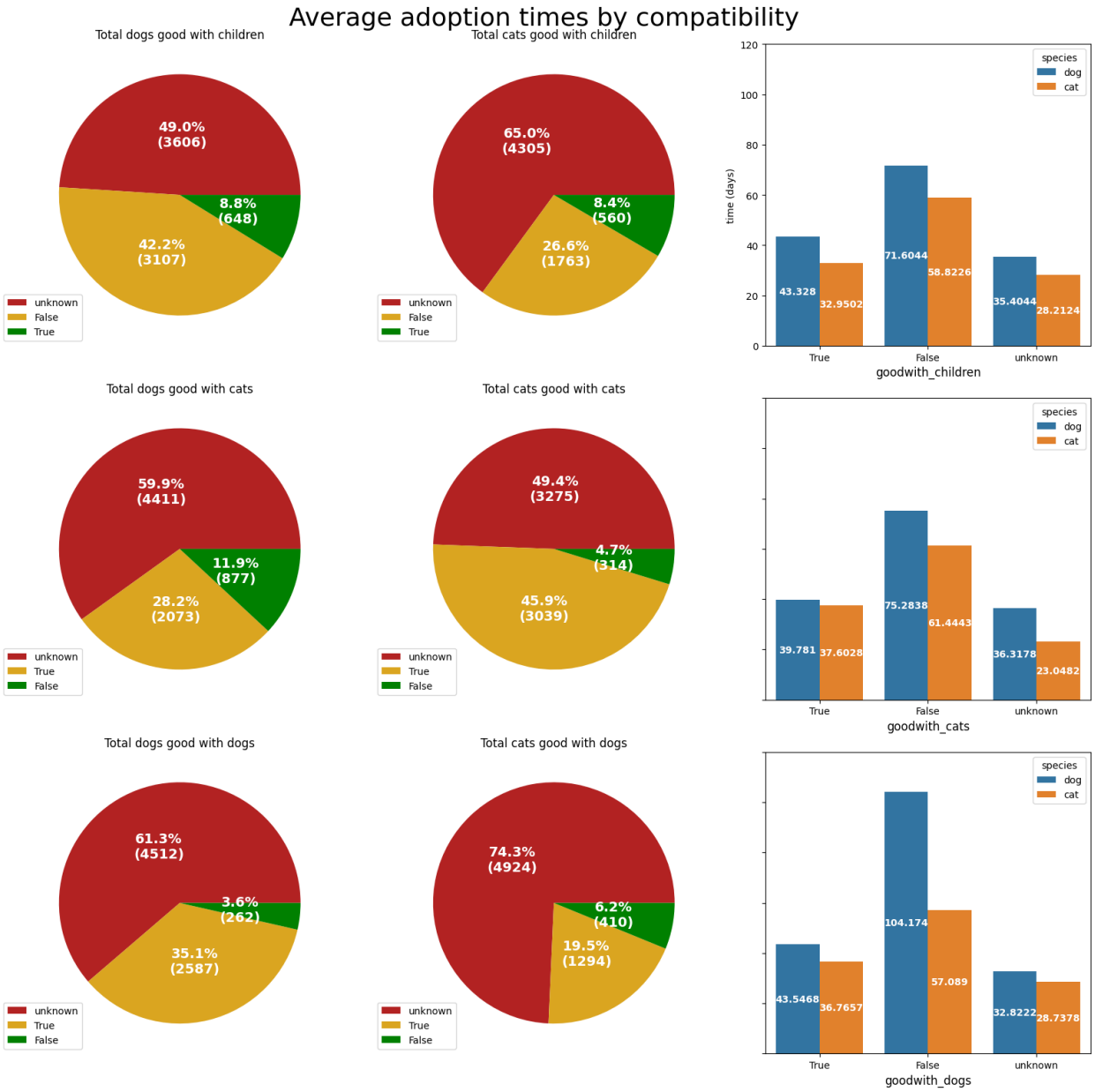


Size is a characteristic that is more meaningful when considering dogs than cats. The size differences in cats are much more subtle until you get into the extra large category, which are considerably harder to adopt. I'd be curious to know what breeds are included in extra large and how much crossover they have with the large category.

The dogs have a fairly even split between small and medium sizes, and then the majority of the remaining animals are large. The bar graph indicates that the size of the dog does matter in adoption times; the smaller the dog the faster it will be adopted. I would be curious to see if this depends on location; I would imagine that smaller dogs would be more prevalent in bigger cities due to smaller living spaces. Unlike cats, large and extra large dogs have a fairly similar average adoption time.



Interestingly, the male/female split is almost the same for dogs and cats, with male animals outnumbering females. Also, the average adoption times between males and females for each species differ by the same amount (3 days) but in the opposite direction from each other: female dogs and male cats tend to be adopted faster than male dogs and female cats. Based on my personal knowledge of these animals this fact doesn't surprise me.



There’s a lot going on here, but the main takeaway is that listing an animal as not being good with children, cats, or dogs hurts that animal’s chances of a fast adoption. This is especially true with dogs who are not good with other dogs. From a marketing perspective it is better to not include this information in an animal’s profile that is not good with any of these categories.

Important Features

Dogs	Cats

• gender • size • coat • distance • spayed_neutered • house_trained • special_needs • shots_current • breed_primary • breed_mixed • color_primary • goodwith_children • goodwith_dogs • goodwith_cats • hasimage • hasvideo • city • population	• age • breed_mixed • breed_primary • city • coat • color_primary • declawed • distance • gender • goodwith_cats • goodwith_children • goodwith_dogs • hasimage • hasvideo • house_trained • population • shots_current • size • spayed_neutered • special_needs

Because all of the petfinder features ended up being selected, this step ended up being fairly meaningless. If I were to refine this model I would start by adding more features so that feature selection could have been more selective.

Models

Because I separated the dog and cat data I ended up using different models for each.

	Dogs	**Cats**

Model	XGBoost	GradientBoostingRegressor
Features **Tuned**	'objective': 'reg:squarederror', 'base_score': 0.5, 'booster': 'gbtree', 'colsample_bylevel': 1, 'colsample_bynode': 1, 'colsample_bytree': 1, 'gamma': 0, 'gpu_id': -1, 'importance_type': 'gain', 'interaction_constraints': '', 'learning_rate': 0.300000012, 'max_delta_step': 0, 'max_depth': 6, 'min_child_weight': 1, 'missing': nan, 'monotone_constraints': '()', 'n_estimators': 26, 'n_jobs': 8, 'num_parallel_tree': 1, 'random_state': 0, 'reg_alpha': 0, 'reg_lambda': 1, 'scale_pos_weight': 1, 'subsample': 1, 'tree_method': 'exact', 'validate_parameters': 1, 'verbosity': None	'alpha': 0.9, 'ccp_alpha': 0.0, 'criterion': 'friedman_mse', 'init': None, 'learning_rate': 0.1, 'loss': 'ls', 'max_depth': 2, 'max_features': None, 'max_leaf_nodes': None, 'min_impurity_decrease': 0.0, 'min_impurity_split': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 233, 'n_iter_no_change': None, 'presort': 'deprecated', 'random_state': None, 'subsample': 1.0, 'tol': 0.0001, 'validation_fraction': 0.1, 'verbose': 0, 'warm_start': False

Performance

Predictions vs Test Values

Average time before adoption

	Predicted	Actual
Dogs	31 days	32 days
Cats	24 days	24 days

Metrics

	R^2	RMSE	Mean Absolute Error
Dogs	.11	41.89	26.97
Cats	0.04	34.78	22.83

Neither model performs exceptionally well. This could be because it's a regression problem. If I had converted the dependent variable into ranges and turned this into a classification problem then there would have been more opportunity to improve performance without getting too complex. Additionally, because I primarily used the animal data, and there was a limited amount of that, all of the fields were deemed important. If I had brought in more location data, like city demographics, or engineered more features then the model would have gotten to actually select features instead of needing all of them.

Further Research/Client Recommendations

Improvements can definitely be made. Rather than making business recommendations to the client I would present them with these ideas on model improvement:

- I decided to leave the dependent variable continuous because it's time data, but converting that to a set of timeframes, and therefore the problem to classification instead of regression, likely could have produced better results. Even just converting the dependent variable from days to weeks could have made a decent impact.
- In this project I opted to not use image data and instead make a column indicating whether or not image and video data existed. If I were to add some image processing into the mix then I probably could have improved on the performance; logically speaking, anyone looking for a new pet is likely going to choose one they find cute, however subjective that is. Therefore, the existence of an image is not as important as what that image is on an online profile.
- Rather than using only the animal characteristics as model features I could have included more city data. For instance, we know that each city has a most adopted breed of cats or dogs. That can be expanded to identify if other characteristics are more prevalent per city, such as coat, size, or color. A model using demographic data would put more weight on the location from which an animal was adopted for its predictions, providing the model more features to choose from.