Austin Animal Center Outcomes

Figs: concise plot to portray most information
Plotly: gif or link to nb

Patches was a real dog, an injured 6-year-old white pit bull, who found herself in the care of Austin Animal Center over 9 years ago. After over 5 years of waiting, finally, she was adopted. While her story may be a success now, while she stayed in the shelter, other animals were denied entry–there just wasn't enough space.

Austin, Texas is a "no-kill" shelter city. This means that they do not put the timer until euthanasia upon their animals–in fact, euthanasia is relatively quite rare at the Austin Animal Center. While this on the surface is wonderful, unfortunately it means that certain animals can stagnate within the shelter, taking up space needed for new intakes–who are therefore turned away. In order to make as large a difference as is needed, and help as many animals as possible, Austin Animal Center is tasked with adopting animals out as quickly as possible.

Adoptions are complicated and often quite random; however, there are certain trends that we can observe. Pit bull terriers often have a difficult time getting adopted, as do senior animals. Young kittens and puppies typically have very brief shelter stays before being taken to a new home. But beyond these most obvious trends, it can be difficult to predict which animals are most likely to either receive a negative outcome (typically either death or euthanasia) or end up staying in the shelter for many years, such as our friend Patches.

Cleaning —

In an attempt to address this difficult question, I wrangled some datasets from Austin Animal Center. They had separate data collections for intakes and outcomes, both ranging from 2013 through the spring of 2025. Therefore, much of my cleaning was done with both datasets simultaneously, but independently. The columns from the intakes dataset were: "Animal ID", "Name", "DateTime", "MonthYear", "Found Location", "Intake Type", "Intake Condition", "Animal Type", "Sex upon Intake", "Age upon Intake", "Breed", and "Color". The data of outcomes had: "Animal ID", "Name", "DateTime", "MonthYear", "Date of Birth", "Outcome Type", "Outcome Subtype", "Animal Type", "Sex upon Outcome", "Age upon Outcome", "Breed", and "Color". Each set had around 173,000 rows, with the intake data including roughly 200 more entries than the outcome data. This makes sense, as those 200 animals are likely those in shelter currently.

I dropped exact duplicates and checked out nulls. There were about 17,000 duplicated animal IDs in both intakes and outcomes, which represent returning visitors. I

also standardized column names and renamed some to represent which dataset they came from (for example, “IntakeSex”, “OutcomeAge”). All datatypes were object, including those that I would have preferred not be (“DateTime”, “MonthYear”, “BirthDate”, and the age columns). To deal with this in part, I turned the “DateTime” columns into actual datetime objects, and culled “MonthYear” as then a redundant column.

I then went through columns one-by-one, cleaning and wrangling as it was needed. “IntakeType”, for example, was left alone, but I consolidated sparse categories within “IntakeCondition”, and culled “FoundLocation” (it just was not within the scope of this project). As far as the outcome columns went, there was some work to be done in the “OutcomeType” and “OutcomeSubtype” columns. In “OutcomeType”, there were some odd categories which seemed to be similar or the same as others; for example, “Rto_adopt” (“Rto” stands for “Return to owner”), which was assimilated into the larger pre-existing category of “Return to Owner”, and “Disposal” (which ended up referring to animals which were a rabies risk), which was assimilated into “Euthanasia”. I made these decisions by cross-referencing with “OutcomeSubtype”. I grouped “Missing”, “Stolen”, and “Lost” together under “Missing”, and lumped “Relocate” and “Transfer” together. For the “OutcomeSubtype” column, I remapped abbreviations, industry slang, shorthand, and other inconsistencies into their associated larger categories. I did find that there were lots of very specific categories with lots of missing values, which made me wonder whether this column would ultimately be helpful. For the time being, I left it in the dataframe. The last column exclusive to the outcome dataset was “BirthDate”, which I simply converted to datetime for the time being.

As for those columns which appeared within both dataframes, several did need some cleaning. The “Name” column, for example, contained many nulls. While in the process of investigating (and also giving into curiosity), I did accidentally print every name, of which there were 50,000. Apologies to anyone scrolling through this version of this project’s cleaning notebook. However, investigating these did give me enough insight to remove asterisks and unnecessary punctuation or spaces from many of these names. I did plan on keeping this column for storytelling purposes later. “AnimalType” contained “Dog”, “Cat”, “Bird”, “Livestock”, and “Other”. I did split the “Other” category into three: “Wildlife”, “Rabbit”, and “Other”, which cut down on the number of unspecified creatures. “Rabbit” was created not for an overabundance of rabbits, per se (there were about 1,000), but for an overabundance of breeds. There were quite many rabbit breeds, so in order to clean up and clarify the “Breed” columns, I separated their animal type. “Wildlife” did end up being a fairly well-populated category, so I was particularly satisfied with that. This did involve cross-referencing “Other” animals with their associated breeds manually (with some help from ChatGPT). I applied these changes to both datasets.

The sex columns (“IntakeSex” and “OutcomeSex”) showed that many animals seemed to have been “fixed” (neutered or spayed) while in-shelter, which was very promising. There were only five categories (“Intact Male”, “Intact Female”, “Spayed Female”, “Neutered Male”, and a few “Unknown”s) and not enough “Unknown” entries for me to be concerned at this point, so I left them be for the time being.

The age columns (“IntakeAge” and “OutcomeAge”) were a bit more complicated, as they were comprised of entries such as “4 months”, “5 weeks”, or “1 year”, none of which parse particularly well for machine learning. For these, I had to convert these strings to numerical year values with a custom function. I then rounded to the nearest integer for simplicity. I also binned these into a new “AgeCategory” column with the categories “Infant”, “Young”, “Adult”, and “Senior”.

The “Breed” columns were also quite complicated. They consisted of over 3,000 breeds each, including many mixes and many obscure entries. I first checked for case-related “misspellings”. Then, I created a new “parse_breeds” custom function, which would divide all entries into “PrimaryBreed”, “SecondaryBreed”, and “IsMix” columns, identifying mixes by slashes and the word “mix”. This left me with about 400 primary breeds and roughly 200 secondary, which was obviously far better than the original 3,000. I also checked through this new list of breeds for inconsistencies, misspellings, and other such manually, by each animal type, and mapped these to correct values with a “replace_breeds” custom function.

The “Color” columns followed a similar workflow, with the removal of secondary colors and manual remapping of colors to make more sense and stay more consistent (there were upon starting about 660 color categories). The only “/” color I kept was “Black/White”, since that is indeed a major color scheme.

Finally beginning to wrap up my cleaning, I revisited nulls, filling those within the “Sex” and “Age” columns with “Unknown”. Most of the nulls, however, were in OutcomeSubtype; in fact, there were quite few otherwise. I dropped this column, deciding with 90,000 nulls, it would not be as helpful as I needed it to be. The “AgeYears” columns were filled with the median value, and the “OutcomeType” nulls were lumped into the “Missing” category.

It was at this point that I at last was able to join the two dataframes, on “AnimalID” and the datetimes for intake and outcome. Post this, I had to drop redundant columns, particularly the “IntakeAge” column. I fixed some weird birthdays and checked particularly old animals for validity, renamed the remaining columns to better fit and reduce confusion, and saved both a clean version of each and the cleaned and combined version.

At this point, finished with cleaning, I wanted to create columns for animals which had been fixed in-shelter, time in shelter, sex, separated from fixed status, outcome success, basic color category, and simplified breeds. I particularly wanted to investigate outcome by intake condition, breed, age, and sex, length of stay by intake type and

condition, breed, age, and sex, and seasonal trends in intake and outcome types, as well as seasonal trends in adoptions.

Finally, after lots of cleaning and preliminary investigation, I began my full EDA process. This began, of course, with the standard basic surveying (describe, head, summary, nulls, and other such). I imported both the separate (cleaned) intake and outcome datasets (just in case), as well as the full combined, cleaned dataframe.

I first filled missing names with “Unnamed”, which was a simple affair, and then began feature engineering.

The first feature created was a “FixedInShelter” column, representing whether an animal entered the shelter “Intact” and left spayed or neutered. This involved checking how many intake and outcome sexes didn’t match, as well as making sure that the gender (male versus female) remained the same throughout. This was accomplished with another custom function. I also created a simplified “Sex” column, in case I needed to reduce dimensionality by just a little bit. This was also accomplished via a custom remapping function, and pulled from “OutcomeSex”.

The next column created was “VisitLength”, representing the amount of time an animal spent in-shelter. This was simply “OutcomeDT” minus “IntakeDT”, which became a (slightly awkward to handle) timedelta. I checked outliers and anomalies, and investigated by creating a further “VisitYears” column (transforming the timedelta to years). Filtering for stays over 3 years, the entries seemed legitimate. Many were pit bulls. I also checked out very short stays, under 5 minutes. Most of these outcomes were “Transfer” or “Euthanasia”, probably situations which occurred or were discussed in-field or prior to check-in, and likely emergencies. These animals, I imagine, either didn’t enter the shelter before being transferred, or were recorded with inexact entry and exit times due to employees being busy or dealing with an emergency.

The basic color column I created (“ColorGroup”) cut down the number of colors within the dataset from 59 categories. This involved a manual remapping by specie, and then another custom function to apply the remap dictionaries and provide debugging feedback. Only 1 “Unknown” remained by the end, with no missing values.

The penultimate feature engineered was “OutcomeSuccess”, which mapped positive/neutral/negative outcomes (in case that would be easier to model than specific outcomes). “Adoption” and “Return to Owner” were positive, “Transfer” was neutral, and “Euthanasia”, “Died”, and “Missing” were negative outcomes.

The final created column was a “Returned” boolean column, representing whether an animal had appeared previously within the dataset. This used “AnimalID” and “Name” and was another measure of success (after all, if an animal returns to the shelter, has its adoption truly succeeded?).

I also reordered all of these columns and those from before for intuitive viewing and interpretation.

EDA —

My univariate analysis ended up longer than it likely needed to be (as did much of this project, in all honesty). “AnimalID” and “Name” were skipped, as they were both not going to be part of the model and comprised of mostly unique values.

“AnimalType” was given a plot of value counts. Mostly it portrayed dogs, secondarily cats, few wildlife, and very few from the other categories (rabbits, birds, livestock, and “Other” creatures). I created a function to print the percentage each of these lesser categories makes up of the dataframe: wildlife ended up being around 4%, while everything else was under 1% of the data. “Livestock” consisted of a whopping 33 entries. Here, I took a moment to return to the main point of this project: helping shelters get animals adopted. Wildlife, then was obviously disqualified. As the other three categories were so very small, I did not believe they were going to give enough information to provide an effective algorithm, and feared they would end up being simply noise. So, to avoid spending further hours on categories that would not ultimately be helpful, I cut them from the dataset. This left the number of entries in the dataset at around 160,000. A new count plot set dogs as the most common animals, but also relatively close to cats in number. There were about 93,000 dogs, and about 68,000 cats.

The “Breed” column was rather complicated. I first separated cats and dogs and counted the breeds; for dogs, there were 196–too many. The value counts for the top 100 dog breeds ended with categories of around 50 entries. There were 103 breeds with totals over 50 entries, and 82 breeds with over 100 entries. I checked which breeds would be excluded if including only the top 100 (there were 21). Moving to the cats’ part of the dataset while I decided, I isolated their breed rows as well and got the value counts. About half of the breeds had over 10 entries, out of a total count of 42 breeds. This meant 22 breeds with 10 or more entries, which meant many were still very sparse and the distribution was dominated by a very few breeds. I created a function to take a dictionary of “specie : threshold”, and convert breeds which don’t meet that threshold to “Other”. About 2600 replacements were made total. 91 dog breeds remained afterward, and about 11 cat breeds. To graph these, I took only the top 15 breeds for dogs (cats already consisted of fewer than that). The dog breeds were relatively spread out but dominated by the American Pit Bull Terrier, Labrador Retriever, Chihuahua, German Shepherd, and Australian Cattle Dog. Likely, the most common breeds are “go-to” breeds when shelter employees are uncertain what a dog’s breed actually is. Cats were almost exclusively domestic shorthairs. Cat breeds are less common and a bit more difficult to parse than dog breeds, so this made sense.

The IsMix column showed about half as many purebreds and mixed breeds–and likely, those purebreds were mostly cats. It was a simple boolean column, so beyond graphing value counts, it wasn’t particularly complicated to evaluate.

Color, however, was a different story (much like Breed). I used an adapted version of my breed remapping function to change rare colors to “Other”, separating dogs and cats again to do so. My threshold this time was 100. 663 replacements were made across both species, most of which were cats. Each specie had 27 colors left after. The dogs’ plot revealed an overwhelming domination of 5 colors: white, black/white, black, tan, and brown. The cats’ plot showed brown tabbies occurring at least twice as often as any other color, although the other colors were a bit more spread out.

The ColorGroup column was significantly easier, having already reduced the number of rare entries. There were 12 colors associated with dogs and 18 with cats (likely because I left the tabby versions of colors in, since tabby is such a prevalent color pattern for them). I dropped 1 unknown. The dogs’ graph showed the same 5 colors dominating, but all colors had over 1000 entries, which was fantastic. The cats’ graph was more spread out, but there were several tiny columns (although there was only 1 category with under 100 entries, thankfully). By far, they were still brown tabbies, and then black, orange tabbies, and black/white animals.

IntakeSex showed via value counts about equal proportions of intact to neutered or spayed animals across sexes. There were slightly more males in both categories as well. I had about 5500 unknown entries remaining, most of which were cats (by an overwhelming majority). Most of the dogs which were “Unknown” were very young, which made sense to me (as very young animals can be difficult to sex until they age). The graph confirmed the data from my value count analysis, with mostly intact animals and males and females being relatively even (although there were slightly more males both intact and fixed). I also created a graph which grouped each sex with its intact and fixed counterpart, for easier parsing and to compare how many males versus females had been fixed. There were many more intact animals in this column.

OutcomeSex showed the very opposite, as far as intact to fixed went: this time, there were far more spayed and neutered animals than intact animals.

For my ShelterFixed column, I didn’t feel it necessary to graph, since it was such a simple binary column. There were about 91,000 animals fixed in the shelter and about 70,000 not (although that includes animals which had been fixed before).

I did, however, graph the BirthDate column, to examine when the most animals were being born. I grouped by months, finding a long tail from around 1992-2008 or so (which I believe is a result of the dataset beginning in 2013 and only having a few animals that old at the beginning). To make the graph easier to read, I added gridlines and cut off that tail, and a seasonality trend became clear. There was a very sharp spike in spring, which makes sense according to animal breeding patterns in general.

AgeYears was an integer-based column of rounded ages. Its mean was just under 2 years, and there was a maximum age of 24. I checked the outliers for sense and validity and capped rows at 20 years, which only affected 8 columns. The graph

showed a very steady decrease in number as age increased, with a long tail along the older ages. There was a very extreme peak at age 0 (animals under a half a year old). The graph of AgeCategory showed the very same thing, with animals becoming sparser as age increased.

IntakeType was dominated by strays, with owner surrenders being a secondary source of animals. There was 1 wildlife entry left, which I needed to check out (seeing as I had eliminated wildlife from the data). It turned out to be a Labrador, so I changed this seeming mistake to “Stray”. There were very few “Public Assist”s, “Abandoned” animals, and “Euthanasia Request”s compared to the main two types.

IntakeCondition showed overwhelmingly “Normal” results, which seemed not terribly helpful. I thought it possibly helpful for storytelling context later, but as far as modeling went, I did not expect it to be of much use. Surprisingly to me, “Neonatal” was a very small category (compared to “Normal”), considering that so very many of the animals taken in were under half a year old.

“IntakeDT” (intake datetime) was graphed via month again, and again showed seasonality. There were lots of intakes very shortly after birthdate peaks appeared (in the late spring/early summer), which makes a whole lot of sense. There was an interesting definite decrease in intakes after COVID hit. I would have expected the opposite–I remember hearing that shelter adoptions were up during the pandemic, since people were so lonely and in need of companionship; the data here, though, shows otherwise.

“OutcomeDT” (outcome datetime) was also graphed by month, and showed peaks in the middle of the year, during the summers. There was a steep drop in 2020 (again, when COVID hit), corresponding with the drop in intakes which occurred around that time. Since then, it has been recovering–but it has still not nearly hit the same levels as before the pandemic.

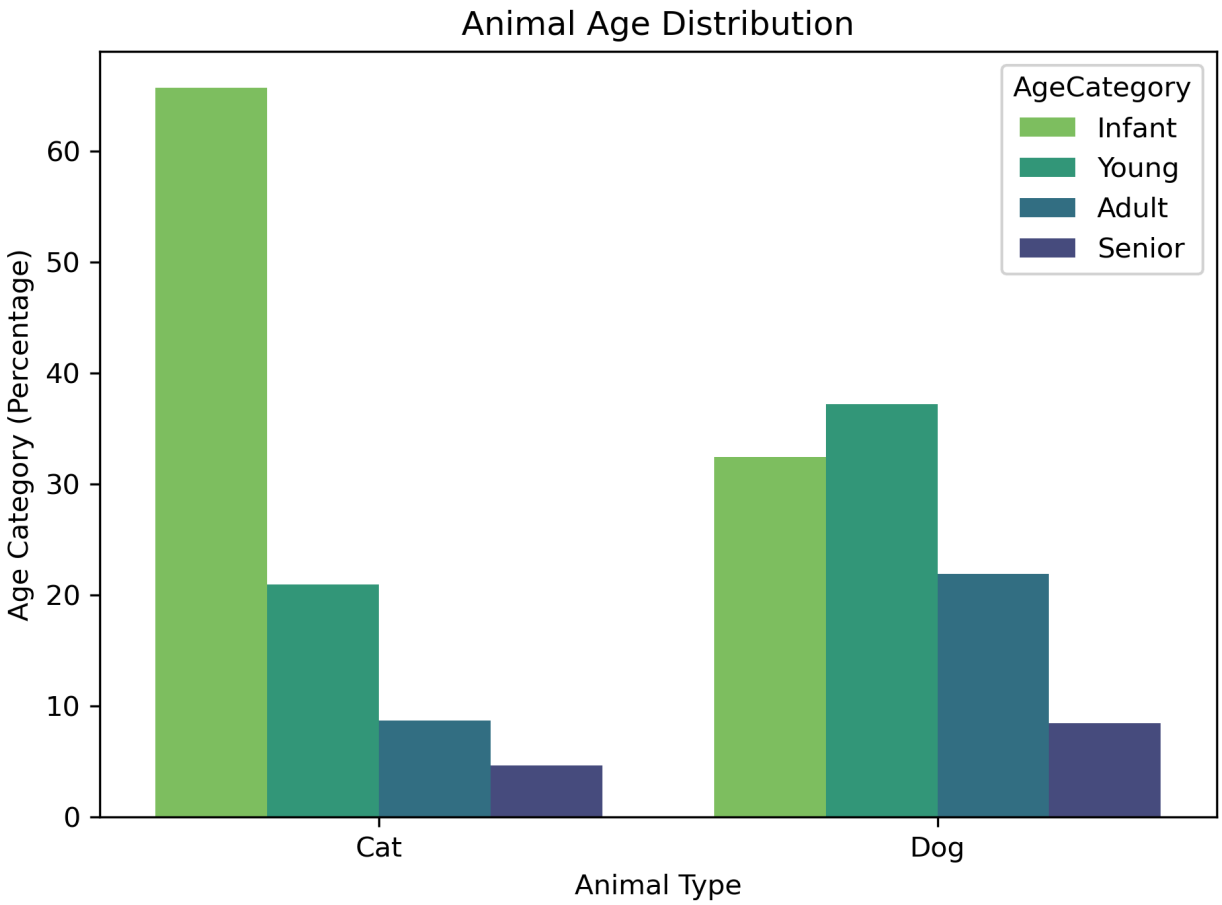
VisitLength was a numpy timedelta datatype, which made it slightly awkward to deal with. It was graphed by days, with a very long tail toward longer stays. Since outliers are some of the most important entries for this project, I did not change them and instead plotted lengths on a log scale. This resulted in an interesting shape with multiple peaks: one at under a day, one at about a day, one at around a week (the largest by far), and then still a long tail trailing off.

“Returned” was again a very simple Boolean column, which I merely displayed value counts for. There were 17,000 return visits from animals (about 10%, which is actually a bit higher than I had expected).

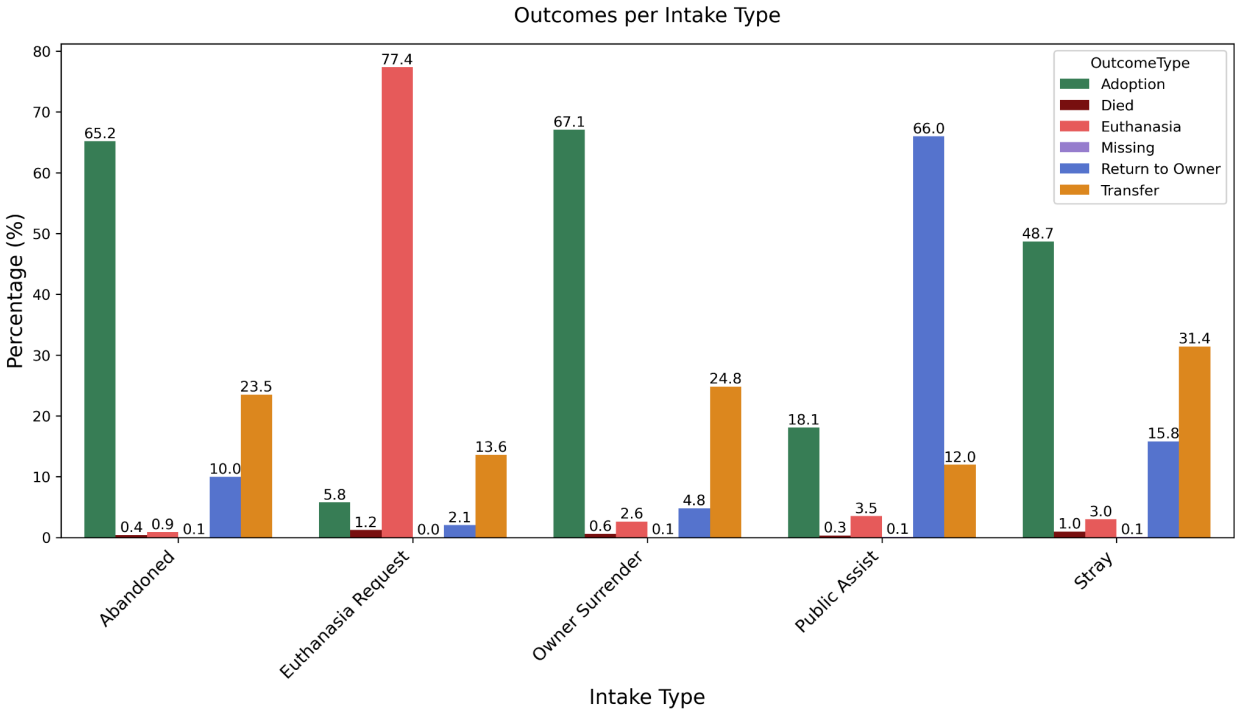
Finally, I examined the outcomes. OutcomeType was dominated by adoptions, almost twice as often as the second-largest category (transfers), which was also about twice as often as the third-largest (returns to owner). Euthanasia, death, and missing entries were quite rare, with “Euthanasia” accounting for a little under 5,000 entries. While this is a lot of lives, in a dataset of 160,000, it is dwarfed by the number of

adoptions. This is due to Austin's "no-kill" policy, which I mentioned above. Austin Animal Center does not euthanize animals without an extraneous reason, such as extreme illness, severe injury, a significant bite history, or other extreme circumstances. Even animals intaken as a "Euthanasia Request" were not always euthanized. In this vein, the "OutcomeSuccess" column agreed overwhelmingly, with over 108,000 positive outcomes and under 6,500 negatives. Again returning to the point of this problem, however, the goal here is not to reduce the number of negative outcomes at the Austin Animal Center; rather, it is to speed along adoptions by recognizing those animals most at risk of negative outcomes or long stays, so that more animals can be intaken, cared for, and also found a home. Something this dataset does not measure is how many animals are rejected for a lack of space, but we can infer it in part from the IntakeDT and OutcomeDT graphs–when outcomes become scarcer, intakes followed suit, as turnover slowed.

Multivariate analysis revealed not only some interesting relationships, but some surprising non-correlations as well. The categories I expected upon going into this to be the most useful were AnimalType, Breed, IsMix, ColorGroup, OutcomeSex, AgeYears, AgeCategory, VisitLength, OutcomeType, and Returned. For the preliminary plots, I did exclude columns with too many categories to be plotted succinctly (particularly Breed and ColorGroup). I used Plotly to graph these in an interactive and easily-interpreted way, first by raw counts, and then by percentages. My full notes are in the "adopt_EDA" notebook, but as they are quite extensive, I will focus on a couple of the percentage-based graphs' highlights here, as I believe those were the most revealing.



I found that cats were overwhelmingly very young (under 1 year of age), whereas (by percentage) dogs had roughly half that many infants. However, dogs had about twice as many (again, by percentage) young, adult, and senior animals, being overall more equivalently spread out than cats. With the exception of infantile animals, both categories followed the same distribution of ages, having more adult animals than seniors and more young animals than adults.



The other graph I will highlight here is one evaluating the outcome percentages each intake type correlated with. In all but two, adoption dominates overwhelmingly; however, two of the five intake types had a different major outcome, both also by extreme degrees. Euthanasia request intakes had a very high rate of euthanasia results (which makes sense; actually, it's interesting how many animals were not actually euthanized here, considering the circumstances of that intake). Public assists had a very high rate of transfer; unfortunately, I'm not wholly sure what that is about. Strays are the only category which didn't have a massively dominating category, although they still had a high rate of adoption. The differences in here made this an interesting graph to view, interpret, and speculate upon, in my opinion.

From here, I took a quick look at the "Unknown" category of the AgeCategory column and the "Missing" category of OutcomeType, as I suspected neither would be helpful when modeling. There ended up being 6 "Unknown"s in AgeCategory, which all had a valid AgeYears entry, so I added in the responding category manually and eliminated all of them. I also checked out "Missing" from OutcomeType, since that category was relatively vague and unhelpful for the goal of the project. It corresponded to 127 rows with fairly average entries, so I dropped this from the dataset. The entirety of the set did still consist of over 161,000 entries.

Next was a lineplot of VisitDays and AgeYears, my two numerical columns. The correlation between the two was weaker than I expected, with a relatively insignificant trend.

Chi-squared and Cramer's V tests were then in order, so I created a custom function once more to produce results from these two and printed feedback. All p-values returned indicating significance; I believe this is less due to legitimate significance and more due to the dataset being quite large. Thankfully, a heatmap of Cramer's V statistics was more revealing. The highest values were columns which were directly related (for example, Color and ColorGroup, or OutcomeType and OutcomeSuccess); however, there were more interesting relationships with lower values which I still found promise in. With just a slight correlation (of values 0.2-0.39), I was interested in AnimalType and OutcomeType/OutcomeSuccess (particularly as outcomes were one of the categories I was focused on), Breed and Returned (as some breeds seemed to have higher return rates) as well as Breed and IsMix (maybe adoption trends were different for a given breed because it is simply more likely to be a mutt), IntakeType and OutcomeType (as intake seems to be affecting outcome, at least to some extent), and IntakeCondition and OutcomeSuccess (for similar reasons). With a slightly higher value of 0.43, OutcomeSex and OutcomeSuccess seemed to have an interesting relationship as well, although I suspected that this was due more to animals being mandatorily spayed and neutered before being adopted out.

Moving on to categories which had been difficult to evaluate via graphs earlier or simply hadn't been evaluated at all, I began with the thus-neglected datetime columns, IntakeDT and OutcomeDT, and did find some interesting correlations and relationships. Dogs had a relatively steady intake and outcome cycle, whereas cats were highly seasonal, having a spike in intakes around spring to early summer and an outcome peak during mid- to late summer. The other particularly notable observation I found was that dogs massively dominated both intakes and outcomes from 2013-2020, but after that, cats became the majority (cyclically, and not by quite as large a margin). It appears (to me) that dog numbers remained low after COVID, whereas cats (while also somewhat lower) largely retained their cycle and extreme spikes.

AgeYears was next, and I did find quite a bit of note in its graphs, although here I will focus on the percentage-based graphs (created with another custom function), since those were the most revealing and helpful. I plotted AnimalType, OutcomeType, OutcomeSuccess, IntakeType, IntakeCondition, and Returned versus AgeYears. After infancy, I found, the percentage of dogs in shelter remained roughly the same, decreasing a bit only in the later years; however, cats spike dramatically in infancy (year 0), then fall drastically, and then slowly grow again as age increases. Adoptions seemed to fall steadily as animals aged, while owner returns increased. Transfers declined slightly as animals aged, but less significantly, and euthanasias increased steadily with age as well. Outcome successes remained relatively steady, although negative outcomes increased a bit as age did, as animals were transferred less and age-related euthanasias became more common. Some IntakeTypes actually seemed to follow a bell curve, interestingly. Strays remained mostly steady, but owner surrenders peaked

around ages 9-11 (sadly, around the age animals become seniors), decreasing on either side of that range. Euthanasia requests increased as age did, beginning around age 9.

For the Breed column, I returned to my top 15 dog breeds dataset from earlier and the separate cat dataset. Cats were relatively simple; there were very few major breeds, with almost the entire demographic being dominated by domestic shorthairs. Maine coons did have a notably high adoption rate. Dogs did reveal some trends, including that pit bulls seemed to have the highest euthanasia rate, and huskies seemed to have the best luck getting positive outcomes. From this, I found that there are indeed differences that seem to be breed-related, though they are smaller than I would have originally expected.

I did create graphs for the Color column, versus both OutcomeType and OutcomeSuccess, but ended up deciding to ditch it in favor of ColorGroup, for the sake of lesser dimensionality and fewer tiny groups. For ColorGroup, I did plot percentage-based graphs again, using my earlier function; they just appeared to be more helpful all-around. I used the same outcome columns to graph against. It still ended up being quite a lot of categories to graph at once, but was definitely more palatable than Color. For cats, the differentiation between colors and tabby colors made it a bit complicated, but I did find some notable relationships and differences. For dogs, there were fewer notable differences that I found.

Returning to OutcomeType, I plotted it against IsMix, Sex, OutcomeSex, AgeCategory, IntakeType, IntakeCondition, and (separately) VisitDays and AgeYears. I decided to stick with OutcomeType, as I did not feel OutcomeSuccess contained enough information to be revealing for my project. I created these plots with raw number counts, as well as percentages, which were yet again more helpful. I did find that infant animals dominated all but two outcome types, including by a significant margin accounting for the most deaths. For the numerical variables (VisitDays and AgeYears), I found that owner returns were most likely to happen within a week, and became far less common after that. Euthanasias usually occurred within 8 days, and also became rarer thereafter. Transfers had a slightly longer typical time range, and deaths even more so. Adoption was most likely between 5 and 45 days–a far greater range than the others. The maximum number of days an animal stayed was near 2,000, but the median was about two weeks. The AgeYears column revealed similar results to what I saw before.

As for VisitLength, I used quite a few columns to graph against, including AnimalType, IsMix, OutcomeSex, Sex, AgeCategory, IntakeType, IntakeCondition, OutcomeType, OutcomeSuccess, Returned, IntakeMonth, and OutcomeMonth. Overall, cats seemed to have a longer stay time than dogs. Their median stay was not much higher than that for dogs, but their third quartile was far higher. Their stays also seemed to range further than dogs; dogs' stays typically were between 3-15 days, whereas those of cats ranged anywhere from 3-34 typically. Surprising me, mixed-breeds actually seemed to have a shorter stay time, which also may just be the result of most

“purebred” animals being cats (since so very many of them are labeled “Domestic Shorthair”). Infants appeared to have the greatest range of stay lengths, whereas seniors’ stays were overall shortest (possibly due to many of these being owner returns or euthanasias). Adoption was actually the outcome type with the largest range and longest stay times, and most outcome types otherwise seemed to be on the shorter side. This makes sense, though; since Austin is a no-kill shelter, the work of Austin Animal Center focuses and relies on adoptions. This goes back to the goal of this project; helping adoptions go quicker, to improve turnover and allow the shelter to help animals it would otherwise have to turn away. Overall, positive outcomes just took longer than neutral or negative ones.

Multivariate analysis finished at last, I moved onto target selection. The goal of this project is to maximize and speed up adoptions, and to this end, I intend to create both a classifier predicting what an animal’s outcome is likely to be and a regressor indicating how long that outcome is likely to take.

The dataset ended with a long list of columns, but I chose as my targets OutcomeType and VisitLength (likely in the form of VisitDays, to avoid the ongoing numpy timedelta issue). Of the rest of my columns, I decided that the most valuable columns to include were AnimalType, Breed, ColorGroup, and AgeYears (which could possibly be replaced to lesser effect with AgeCategory if needing only categorical columns). IsMix did not have enough information on its own or show enough significant relationships to warrant its place, Color simply had too many entries (and is replaced by the superior ColorGroup column), IntakeType and IntakeCondition did not seem to have too great an effect, and Sex did not seem to have as significant an effect as I suspected, either, and OutcomeSex appeared to be more the result of outcome than a cause. I worried that the datetime columns (IntakeDT, OutcomeDT, IntakeMonth, and OutcomeMonth) would be too difficult to include in this already-mixed pair of algorithms, OutcomeSuccess did not have enough detail to be my target, and Returned did not contain enough information to be included, either.

I was finally able to reorder the final dataset and save it! As a summary, I reprinted the value counts of each categorical variable. At this point, my suggestions for the shelter would be to set a list of breeds for employees to select from when entering data, possibly include primary breed and secondary breed columns (and/or a mixed-breed indicator column), as well as some color standardization. My mentor suggested for this notebook that I skip the long outputs, which I agree with and hope to change in the future.

Preprocessing —

Upon moving into preprocessing, my columns included OutcomeType, VisitLength, VisitDays, AnimalType, Breed, ColorGroup, AgeYears, AgeCategory, and

IntakeMonth. From these, I dropped AgeCategory because it was extraneous (it had been included in case the models I used could not handle both numerical and categorical data, but after deciding what models I would use, I ended up not needing it. I also dropped VisitLength due to its numpy timedelta-ness, which was difficult to work with, and just renamed ColorGroup to Color for simplicity.

From here, I did one last null check (there were none) and some outlier evaluation. VisitDays, my main target for this, was very heavily skewed. It appeared that after 750 days, though, these outliers seemed to teeter off. The median was 6; however, since extended stays were so critical to the goal of this project, I didn't want to immediately eliminate them. Looking deeper, there appeared to be 30 entries over 1000 days and only 74 over 730 days (two years). Since 74 out of over 160,000 is not terribly significant, I felt okay capping this column at 2 years. This hardly changed the summary stats at all. From here, since the distribution was still so very skewed, I decided to implement a log scale for the algorithms to deal with, then de-log it for interpretation and scoring after predictions were made.

The other outlier-heavy column I looked at was my other target variable, OutcomeType. It was very heavily imbalanced, in favor of adoption. However, in order to avoid losing the significance of the minority (what is really important here), I decided not to balance the dataset for it. I did decide, in the end, to make this a binary classification problem, focusing on the categories "Euthanasia" and "Died". Predicting which animals were most at risk felt the most important, after all. From there, I would implement class weighting measures in the algorithms themselves.

Continuing along my preprocessing, I established separate X and y variables for regressors and classifiers each. Here is where I made the classifiers' y a binary problem focusing on high-risk outcomes. I also label-encoded my classifier y for models which needed numerical labels (several of the ones I would use) and sampled 25% of each variable for preliminary and quick testing.

My other encoding steps included dummy encoding for AnimalType (there being only two categories, after all), accompanied by ensuring the columns matched and applying the same treatment to full and sample variables for each type of algorithm. I used target encoding with cross validation to avoid overfitting for Breed and Color, since I felt there were too many categories here to create dummies of. I fit and transformed training data and only applied the transformation to the test data to avoid problems later.

From here, I double-checked each X variable via the pandas .head() function, applied the log scale to my regression y variables to combat the heavily skewed distribution, and reset the variables for simplicity and consistency later. I also had to adjust the X categorical columns to a categorical datatype for XGBoost and LightGBM, as well as set aside string versions for a CatBoost model.

Modeling —

Finally, I had made it to modeling. Setup involved creating a function for quick tuning via RandomSearchCV, which identified regressors and classifiers, applying negative root mean square error evaluation to the former and weighted F1 scores to the latter. It would return feedback of the best score and parameters, as well as a random_search object. I created alongside this functions to evaluate the regressors and classifiers. To avoid a function which took up three pages, I did separate these into their respective model types. The regressor evaluator returned printed feedback and a dictionary of R2, mean absolute error, root mean squared error, and R2 and RMSE train/test score comparisons and gaps. The classifier evaluator returned similar printed feedback and a dictionary of accuracy, F1 weighted and macro scores, as well as accuracy train/test score comparisons and gaps. I saved these results into lists of dictionaries for later evaluation.

My models included XGBoost, LightGBM, and CatBoost, with a classifier and regressor each. I began with XGBoost, since it felt like the “base” of the three. Its regressor’s best NRMSE score returned was -1.305, which, translated to RMSE, ended up being about 60.4% of my mean, and I then created and fit a model with the returned best parameters from my quick tuning function. I saved the resulting dictionary into my regressor list. The classifier returned a best weighted F1 score of 0.836, which felt pretty good to me. I followed the same steps: initiate and fit model, then evaluate with my classifier evaluator function. I saved these results into a classifier results list.

For my next model, LightGBM, I followed the same pipeline. Quick tuning my regressor resulted in a best NRMSE of -1.304, still 60.4% of the mean, and the classifier received a weighted F1 score of 0.844. I saved both dictionaries into their respective lists for later, again.

My last model to test was CatBoost. Quick tuning resulted in a regressor best NRMSE of -1.305 (the same as XGBoost, actually, constituting the same percentage of the mean) and a classifier best weighted F1 score of 0.800. I initiated and fit the models according to their best params, ran them through my evaluation functions, and saved the results once more.

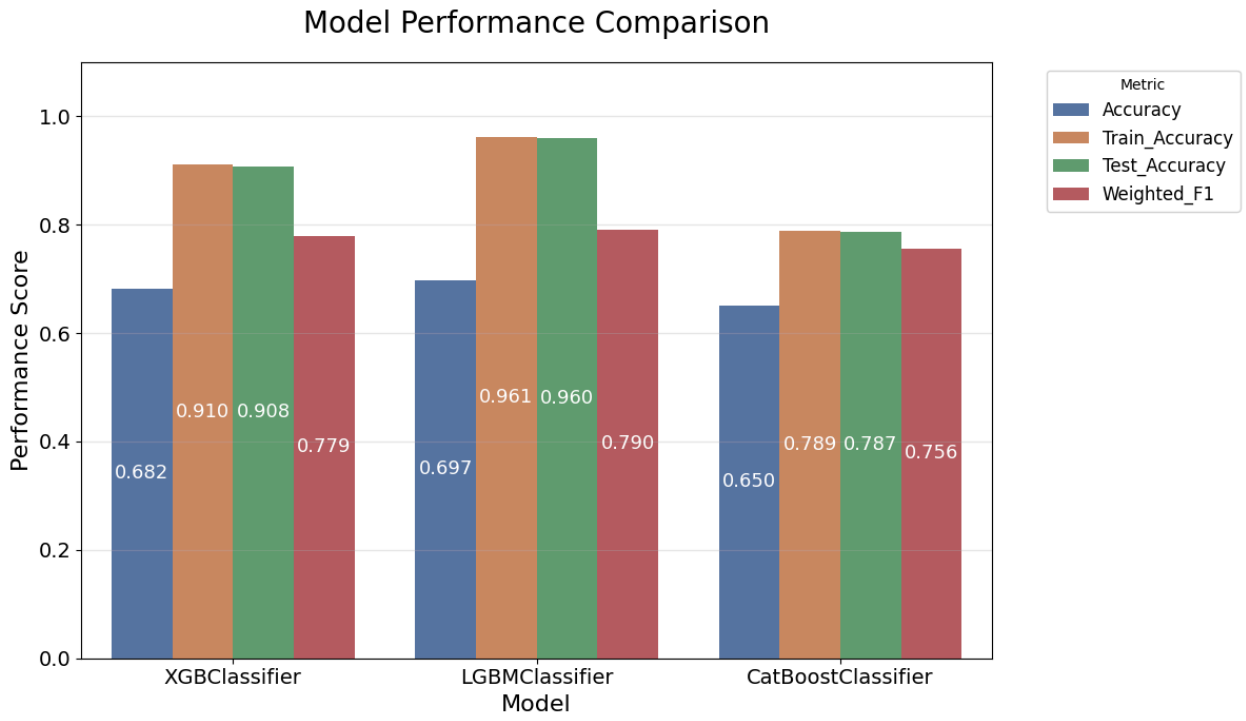
My final evaluation process involved creating some rather long functions which very thoroughly evaluated each model and ranked them based on several criteria. The regressor final evaluation function returned prediction range and feature importance validation, basic metrics, the train/test score gap, as well as a dictionary of all the same stats I originally looked at as well as the new statistics it calculated. The classifier final evaluation function evaluated feature importance, class distribution, and an AUC-PR analysis, as well as basic metrics and test/train gap analysis. It returned a dictionary of all this as well as much printed feedback.

I created separate functions to take these dictionaries and rank models based on several criteria. The regression ranker returned a printed list of models in order from best to worst, along with summary stats (RMSE, R2, train/test gaps, and RMSE percentages of the mean). The classifier ranker printed the same feedback, with corresponding statistics (including accuracy, weighted F1, and an accuracy train/test gap).

The regression rankings came in with CatBoost in the lead, XGBoost following, and LightGBM lowest. Unfortunately, even after all this work and tinkering, the best regressor still had an RMSE over 60% of the mean and an R2 score hardly over 0.05. I believe this reflected the randomness which is associated with adoption trends; unfortunately, I am ultimately unsure if such a relatively simple model is able to predict length of stay well. Given these scores, I decided to leave the regressors behind and focus on the best classifier. I also did not graph these against each other, since the scores were so abysmal.

After applying the same evaluation functions and saving the results of the classifiers, my classifier ranking returned with far more promising results. I waited to rank these until I completed a couple extra model selection steps.

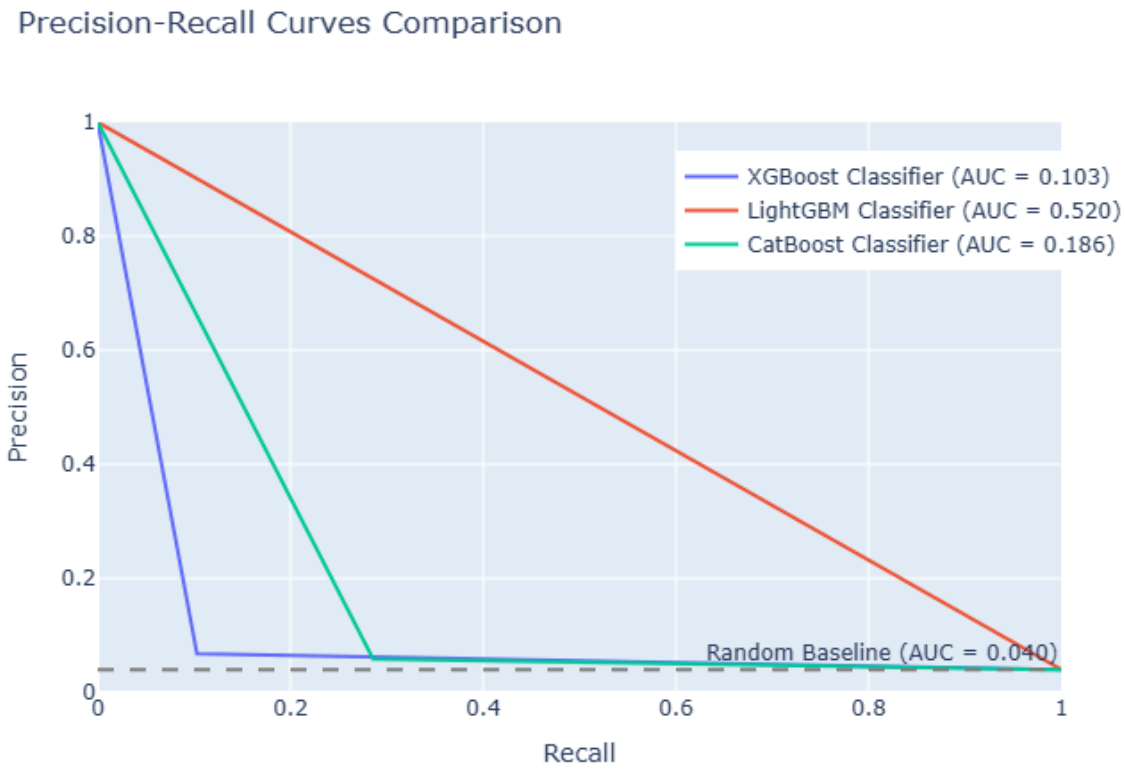
I created a dataframe of the results to graph accuracy, train accuracy, test accuracy, and weighted F1, as well as PR-AUC graphs.



The scores appear pretty similar in trend, but LightGBM dominated in every metric, making it a very clear winner. In fact, its train and test accuracies were above 90. I was not very sure why these were so different than its general accuracy score (a little

under 70%), but seeing that they were similar to each other made me a bit less concerned about it.

I calculated y probabilities and graphed the models' PR curves together as well.



Once again, LightGBM dominated the field, and not by any small margin. Though the PR curve was oddly straight, it beat the competition by a landslide.

Finally, I did use my classifier ranking function. The ranking results put LightGBM in the lead in every listed metric. I was surprised that CatBoost did so poorly, coming in last, as that is what I expected to perform best. However, with a relatively high accuracy and weighted F1 score, a significantly better PR curve, and all-around better results, the LightGBM model was the best model.

From here, I initiated my very last step: final tuning. I actually did this without a custom function (gasp). I initiated a basic model, created a scorer, and initiated a final random search, more thorough than before searches had been, and implementing the full dataset. The final model was saved as 'lgbc_final' and then run through the same final evaluation function I had used earlier. The results were similar, if a tad improved, and I suggest going to the notebook (**LINK**) to view those.

Summary —

The goal of this project was to compile two models together in order to predict both the outcome a shelter animal was likely to receive and how long that outcome would likely take. Unfortunately, the regression models employed for the second part of that question did hardly more than random guessing, so in the end, I decided to leave them behind The classification models, however (predicting what outcome an animal was likely to receive) did fairly well. All said and done, I chose a LightGBM Classifier ('lgbc_final') as my final model, as it significantly outclassed both other models (an XGBoost Classifier and a CatBoost Classifier). I am satisfied with the final model's performance, especially given the very random nature of adoptions (as witnessed by the poor regression models).

I believe that this, or something similar, could help shelters (in this case, Austin Animal Center in particular) predict which animals need extra help to be adopted and which don't, thereby increasing efficiency within the shelter and increasing turnover, which is necessary for a shelter to continue taking in pets.

Some items of note and things I would change or suggest:

- I did not include every variable possible, nor did I combine variables into more complex categories (e.g., compiling age + breed into a column which has categories such as "4 year old pitbull").
- I did not use timestamped data, merely the intake month and time stayed.
- I am uncertain whether a regressor model for this problem will ever be feasible, but I hope that someone more experienced than I is someday able to figure it out.
- I limited the animals I evaluated to dogs and cats, and ignored other species (to avoid introducing extra "noise" into the algorithms and due to the fact that most of them were out of the scope of this project, particularly wildlife).
- I focused on a binary problem for the classification, that being a positive class consisting of the worst outcomes ("Euthanasia" and "Died") and a negative class of everything else (including not only "Adoption" but outcomes such as "Return to Owner" or "Transferred").
- There are certainly trends in animal adoption; however, there is also a very large amount of randomness, due to the human nature of this. Adoptions are often not an analytical or straightforward decision, but more emotional, driven by either pity, loneliness, or just the desire to take care of something or have an ever-present friend. Because of this emotional bias, adoptions will likely never be perfectly predicted; however, I did my best by this shelter, and I feel fairly good about my results (for now).

As for myself, I hope to come back to this and:

- Merge my pre-processing steps between classifier and regressor, separating variables later to avoid redundancy
- Reduce the over-complications present throughout the notebooks
- Keep my pre-logged VisitDays data as well as the logged version, to determine whether my models can handle the skewedness on their own, and overall adjust my handling of the log
- Explore feature importances further
- Revisit the issue of balancing or leaving OutcomeType, as well as re-evaluate my focus in regard to that column (possibly focusing on adoptions, instead, since euthanasias are so rare in a no-kill city and I am ultimately trying to speed those up)

Thank you for your interest in this project! I worked long and hard on it, but I am very new to data science. There are many things I could (and probably should) have done differently, and those I will get to learn and revise more as I grow as a data scientist. So rather than being the end-all, I hope this inspires someone more experienced than I to try their hand at this effort--make a difference for somebody.