Aditya Pulipaka
10/07/2025

Project 1: The Power of Pet Adoption Predictors

I'd like to begin with a discussion of the motivation behind creating a machine learning model for pet adoption. Arguably one of the worst outcomes for this business and for the pets involved is the prospect of never being adopted and rather spending a majority of their life in the shelter or without an owner. Although human experts may know whether a certain animal fits their current market (in this example, prospective Austin pet owners), there is a great possibility of bias introduced into the making of this decision. A human decision maker could be tied emotionally to the pet, or they might not consider seemingly unimportant factors, like the time of day when a purchase occurs. With a machine learning model trained on past adoption data, new outcomes could be predicted with greater accuracy. In addition, an example of a model's flexibility is the possibility of using the generated model with various input times of day to find the highest probability that an animal may be adopted (assuming the probability varies throughout the day). This level of convenience and flexibility while having relative confidence in the output provides an attractive addition to human expert decision.

Before training could begin, the dataset had to be cleaned considerably. Throughout this process, `data.info()` and `data.head()` proved to be invaluable for understanding the makeup of my dataset at any given point. To begin with, I dropped the 'Animal ID' column, then removed duplicate rows in the dataset. The 'Animal ID' column was dropped due to its nature as a likely confounding variable. The ID of an animal realistically has no influence on an animal's outcome if no other information is known. Duplicate rows were then removed based on identical data for name, date, DOB, outcome type, and other identifying factors. Next, to consider temporal data in a numeric format, the individual year, month, and hour were extracted from the DateTime column, ignoring date, minute, and second data due to their noisy nature and ignoring time zone since it was either UTC-5 or not present. Then, the month and hour were put in periodic format, transformed with a cosine function. I found this method through some online research on this Nvidia Developer site, which explained how the true behavior of hours in a day or months in a year isn't linear, since hour 0 is temporally close to hour 23 in a day and month 0 is temporally close to month 11. Next, 'Outcome Subtype' was dropped, since it was directly part of the dependent variable, and the description of the project didn't require a prediction of subtype. Then, 'BirthMonth' and 'BirthYear' were extracted from 'Date of Birth' to allow for prediction based on possible preferences of pet birth months or years while reducing noise introduced by date. The 'Name' column was collapsed into a Boolean column expressing whether the animal had a name, a possibly important metric considering that nameless animals mays have different outcomes than named animals. Finally, I dropped all rows that had null 'Outcome Type,' considering that rows without an outcome are useless for training our model.

Next, to prepare the data for classification, I converted categorical variables to one-hot encoding. Firstly, I found the number of unique values for 'Outcome Type', 'Animal Type', 'Sex

upon Outcome', 'Breed', and 'Color'. 'Breed' and 'Color' were incredibly sparse and did not make sense to include, as they would likely contribute to dataset noise. These columns were dropped. Next, 'Type' and 'Sex' were one-hot-encoded while 'Outcome Type' was one-hot encoded then inverted, since the desired outcome was for Adoption to return True while Transfer returned False. This layout fit better with the target metric of $\frac{True\ Adoptions}{True\ Adoptions+False\ Adoptions}$ obtained from our theoretical worst outcome.

Before classification, I performed univariate analysis to catch any last-minute discrepancies between my expectations and the reality of the dataset. Firstly, from `data.describe()`, we see that Birth year extends to 99. Upon some analysis, it is evident that this means 1999 is the oldest birth date for any animal in our dataset. However, this introduces a problem in that the birth year of 99 appears much greater than any other year, so the column was also dropped, maintaining only the Birth Month from DOB info. This made sense since birth year likely has no influence on outcome outside the animal's age, while the month may indicate certain seasonality in adoptions. Next, plotting histograms on animal age and hour of purchase showed clear trends in this data as well, with a great majority of data falling in the afternoon to evening hours of the day and a great number of handled animals being under a year old.

Now that the data was prepared for training, I sliced the DataFrame to place all independent variables in array X and the independent variable OutcomeType in array Y. I then used `train_test_split(x, y, test_size=0.3, stratify=y, random_state=1)` to ensure the same general proportion of adoptions in testing and training data and to guarantee reproducible results with a seed of 1. The first classifier I used was the Stochastic Gradient Descent Classifier from Scikit-Learn. I ran `SGDClassifier(loss="perceptron", alpha=0.05, random_state=1)`, obtaining quite abysmal results with an accuracy of 0.35, a precision of 0.12, a recall of 0.35, and an f1-score of 0.18. This made sense considering that the data for adoptions may not be linearly separable. To support this, the K-Nearest-Neighbors classifier achieved high average scores of 0.84 for each of the four metrics when run without cross-validation and with an arbitrary k-parameter of 3. Finally, utilizing KNN with cross-validation testing 1 – 99 neighbors (inclusive) returned an optimal k-parameter of 2. When optimized for our target metric of precision, it obtained a 0.89 precision score on test data, indicating a rather useful and robust model given the non-critical context of this data. Since this is simply being utilized to decide between maintenance or transfer of an animal, I would feel comfortable relying on this model to make business decisions for the shelter – specifically 89% confident ☺.

As an aside, the linear classification results were much better (near 0.8 for all scores) before I tried making the month data cyclical, but the knn methods improved after conversion. This was interesting and might call for further investigation as to why the conversion of month data caused the overall set to cease to be linearly separable.