Daniel Andrade
Matthew Bundas
Shahriar Dipon

Motivation and Proposal

Austin is a unique city because it maintains a 90% survival rate of animals which enter its shelters. Although most of the animals are adopted, the team was interested in the duration of an animal's stay. For this project, the team used public data sets from the Austin Animal Shelter and machine learning techniques to predict the time an animal spends in the shelter. This investigation was based on the animal's attributes which were provided in the public data sets. These attributes included: age, color, breed, sex, and outcome type.
The team's motivation for this project was to investigate machine learning models that could be beneficial for other shelters with occupancy planning, specifically in determining which animals to transfer to sanctuaries, no kill cities/ shelters, or foster homes. The team also wanted to choose a meaningful and unique public data set that had not been analyzed excessively on other platforms such as Kaggle or UCI Machine learning Repository. Finally, the team hoped that studying this data would have the potential to reduce the unnecessary killing of animals in shelters.

Problem

Use machine learning algorithms and data mining techniques in an effort to predict the time an animal will spend in a shelter based on its physical attributes.

Solution

Acquire, clean, and merge the two data sets described below and apply various machine learning algorithms. Compare and contrast the algorithms performance including accuracy and computational costs.

Data

Both data sets are provided by the City of Austin Open Data Portal which is designed to provide high value data to users interested in finding out more about the city. For this project, we used two data sets. Animal Center Intakes which included data from Oct, 1st 2013 to March 20th 2021. This data set represents the status of animals as they arrive at the Animal Center. All animals receive a unique Animal ID during intake. Annually over 90% of animals entering the center are adopted, transferred to rescue or returned to their owners. This data set contains 124,492 rows and 12 columns which are Animal ID, Name, Date Time, Month Year, Found Location, Intake Type, Intake Condition, Animal Type, Sex Upon Intake, Age Upon Intake, Breed, and Color.
The second data set was the Animal Center Outcomes from Oct, 1st 2013 to March 20th 2021. Outcomes represent the status of animals as they leave the Animal Center. The Outcomes data set reflects that Austin, TX. is the largest "No Kill" city in the country. This data set contains 124,880 rows and 12 columns which are Animal ID, Names, Date Time, Month Year, Date of Birth, Outcome Type, and outcome Sub-type, Animal Type, Sex upon Outcome, Age upon Outcome, Breed, Color.

Data Preparation
Before analyzing the data, the first step is to investigate and prepare the data. As described in the previous paragraph, the Austin Animal Center Intake consists of the following attributes: Animal ID, Name, Date Time, Month Year, Found Location, Intake Type, Intake Condition, Animal Type, Sex upon Intake, Age upon Intake, Breed, and Color. Animal ID is the unique key between both data sets.
The 'Name' column was mostly blank and it is an arbitrary value, so this attribute was dropped, it was not considered in our analysis. 'Date Time' and 'Month Year' were found to contain the exact same information, so 'Month Year' was dropped. The 'Found Location' consisted of specific street addresses inside the City of Austin and almost all the entries were unique, this column was dropped. 'Intake Type' consists of Stray, Owner Surrender, Public Assist, Euthanasia Request, and Abandoned. All these categories were kept. Figure 1 below shows a histogram of this column.

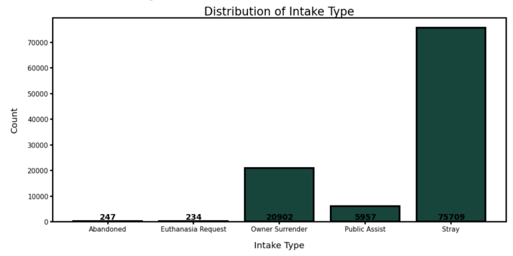

Figure 1. Histogram of Intake Type

'Intake Condition' consisted of the following attributes: Normal, Sick, Injured, Nursing, Aged, Other, Feral, Medical, Pregnant, and Behavior. All these categories were kept and used for analysis. Figure 2 provides a visual representation of the distribution.

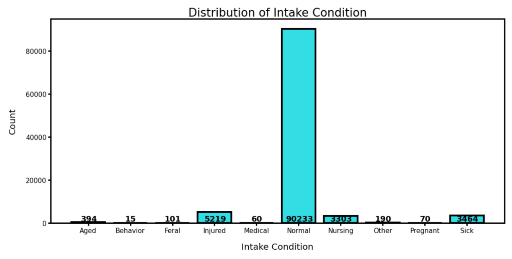

Figure 2. Histogram of Intake Condition

The ‘Animal Type’ attribute was found to contain data for ‘Dog’, ‘Cat’, ‘Other’, ‘Livestock’, and ‘Bird’, the instances of ‘Other’, ‘Livestock’, and ‘Bird’ had very few instances, less than 0.01% of the data, so they were dropped. Figure 3 is a histogram showing the quantity of each animal type.

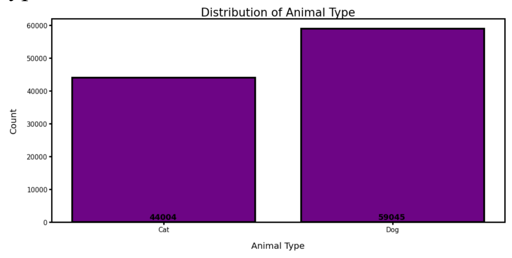

Figure 3. Histogram of Animal Type

‘Sex upon Intake’ consisted of Intact Female, Intact Male, Neutered Male, Spayed Female. and Unknown. All categories were kept for analysis and a histogram is shown below in figure 4.

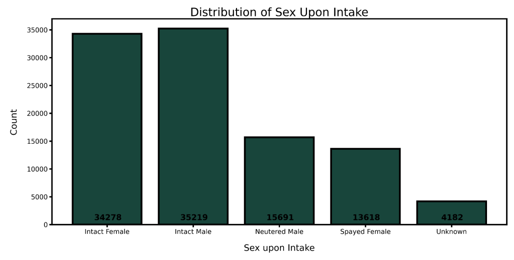

Figure 3. Histogram of Animal Sex Upon Intake

‘Age upon Intake’ was dropped because the animal’s age was also present in the Austin Animal Center Outcomes data set. A detailed description of animal’s age is below.
As we learned this semester, most of the techniques in machine learning have an iterative process. The ‘Breed’ and ‘Color’ columns were initially kept in their raw format, with 2419 and 562 instances, respectively. The raw format was used with label encoding during our initial exploration of the data. The team determined that it would not make sense to use the raw format with One Hot Encoding as it would significantly increase the size of our data set. After running our algorithms with this format, the team sought a reasonable method to reduce the number of unique values in these columns, while also maintaining the quality of information. The hope was that reducing the unique values would increase both our accuracy and run time. The data in these columns used a slash symbol as a demarcation between the specific breeds and colors, so the team reasoned that the first string before the demarcation would be the most dominant breed or color. In performing this reduction the breed was reduced to 646 unique instances and the color was reduced to 59 unique instances. To keep mixed breed animals and multicolored animals separate from pure bred and single color animals, two additional columns titled “Mix Breed” and “Mix Color” were created with binary values. It was seen that this is a significant reduction in the cardinality of these columns, but the distribution showed that some of the breeds and colors had very few instances. To address this issue, the team decided to group these instances as rare/ultra rare breed and rare colors. This reduction resulted in a number of unique breeds to 105 and number of unique colors to 41.

Austin Animal Center Outcomes consisted of the following attributes: Animal ID, Name, Date Time, Month Year, Date of Birth, Outcome Type, Outcome Sub-type, Animal Type, Sex upon Outcome, Age upon Outcome, Breed, and Color. Some of these attributes appear in the Austin Animal Intake data set so the redundant columns were dropped which included ‘Name’, ‘Month Year’, ‘Animal Type’, ‘Breed’, and ‘Color’. The attribute ‘Date of Birth’ is not relevant for our analysis because the animal’s age is already listed, so this attribute was deleted. ‘Outcome Sub-type’ was mostly blank, so this column was omitted from our analysis. ‘Sex upon Outcome’ consisted of Intact Female, Intact Male, Spayed Female, Neutered Male, and Unknown. Because it may be reasonable to argue that a change in Sex while at the shelter, not just the absolute sex, is important, A “Sex_Changed” column was engineered, where animals who have different Sex from Intake to Outcome have value 1, while animals who have the same Sex, have value 0.

‘Age upon Outcome’ had 49 unique values and they were string values with a combination of years, months, or days. To keep units consistent, all values were converted to a numeric value of days. Animals that were listed with an age of “0 years” or any negative value were dropped. Finally, both data sets were merged based on the ‘Animal ID’ attribute to create a working data set for our analysis.While investigating this data set, it was discovered that there were multiple instances with the same Animal ID. Since the outcome of all the instances was “Return to Owner”, only the most recent instance was kept. Figure 5 shows the distribution of Outcome Type. The response variable in this project, the Time in Shelter, was calculated by subtracting the Intake Time from the Outcome Time, and converted to units of days. It was noticed that, due to bad data, several hundred instances had negative Time in Shelter, as the Time of Outcome was earlier than the Time of Intake, likely to do typos. These instances were dropped from the dataset before being used in the model.

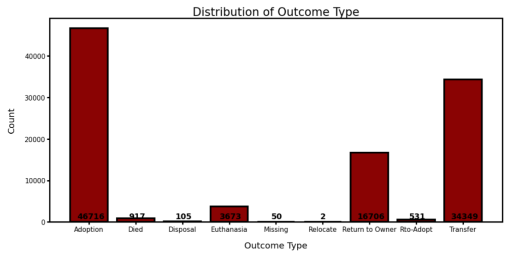

Figure 5. Histogram of Outcome Type

Analysis

Encoding

As described above, the majority of the data is categorical in the form of text strings. Since most of the algorithms in Sci-Kit Learn require numeric data, the data set had to first be encoded with numeric values. Fortunately, the Sci-Kit Learn package has multiple methods to encode data, and two popular approaches are Label Encoder and One-Hot-Encoding. However, each approach has benefits and drawbacks.

The first method used to encode the categorical data was Label Encoder. This method converts each value in a column to an integer. One of the drawbacks of this encoder is that the numeric values could potentially cause an algorithm to misinterpret the data as having some sort of hierarchy or order to the categories/ values. An example of how this works can be seen with the 'Intake Type' column. The figure below shows the contents of the attribute and the output when the Label Encoder Fit Transform is used. It is seen that the value of Wildlife might be weighted higher than any other value in the column. The advantage of this method is that it does not create additional columns like One Hot Encoder. Due to this fact, this method is less computationally expensive than One Hot Encoder.

Original Value	**Label Encoder Assigned Value**
Abandoned	0
Euthanasia Request	1
Owner Surrender	2
Public Assist	3
Stray	4
Wildlife	5

Figure 6. Label Encoder Values

To avoid the issue of weight or hierarchy, another common approach is One-Hot Encoding. Using this strategy, each value of an attribute is converted into a new column and assigned with a binary value 1 or 0 (notation for True or False) to the column. Using the same attribute above and using the One Hot Encoding fit and transform function, the result would be as shown in Figure 7. Though this approach eliminates the hierarchy/order issues, it does have the downside of adding more columns to the data set. In our data set the attributes for 'Animal Breed' and 'Animal Color', after reducing cardinality, contain 105 and 41 unique values respectively. This method of encoding increases the size of our data set enormously and becomes computationally expensive. When using One Hot encoding, and processing the data as described above, we are left with 103,049 instances and 186 columns.

Original Value	**OHE1**	**OHE2**	**OHE3**	**OHE4**	**OHE5**	**OHE6**
Abandoned	1	0	0	0	0	0
Euthanasia Request	0	1	0	0	0	0
Owner Surrender	0	0	1	0	0	0
Public Assist	0	0	0	1	0	0
Stray	0	0	0	0	1	0
Wildlife	0	0	0	0	0	1

Figure 7. One Hot Encoder Values

Dimension Reduction

With such a large dataset needing to be processed, the use of dimensionality reduction was explored in this project, in particular Principal Component Analysis (PCA). PCA attempts to transform the data into a new space with fewer dimensions, enabling learning to be done more efficiently. However, with our data set, PCA did not perform well. Essentially all of the variance is explained in the first principal component, over 99.9999%. When PCA was implored in our models, they trained and predicted quicker, but saw decreased performance, thus PCA was not used as part of our models. Due to the nature of our problem being a regression problem, LDA was not possible, and due to the size of our dataset, other dimension reduction techniques such as Random Gaussian Projection were not practical.

Data Scaling

With the exception of one feature, all of the data was categorical. Age was the only non-categorical feature present in our dataset, and it was normalized using a Standard Scaler.

Regression Discussion and Results

Now that the methods for encoding, dimension reduction, and scaling were determined, the next step was to choose the algorithms for analysis. As with data encoding, and discussed in the encoding section above, there are benefits and drawbacks related to each method. The target for this project is Time In Shelter, since this is a continuous variable, regression techniques will be employed.

Random Forest
The first method for analysis was the Random Forest Regression algorithm. This is an ensemble approach, making use of many individual Decision Trees. Decision trees are very popular and practical for supervised learning due to their relatively quick run time and interpretability. The algorithm starts at the root of the tree and continuously splits the data on features that result in the largest information gain. This ensemble method has the benefit of combining several, smaller models to hopefully increase regression performance. Compared to individual Decision Trees, Random Forest Regressors are better able to generalize, and are less sensitive to outliers.

When dealing with such a large and complex dataset, finding ideal hyperparameters for the Random Forest Regressor was essential. Despite Random Forest Regressors being less prone to outliers and overfitting, performance varied quite a bit with changing hyperparameters. To systematically ensure a reasonable sample of hyperparameters were tested, a grid search with 5-folds was implemented to find optimal values for three hyperparameters, n_estimators, min_samples_split, and max_depth. The value of these hyperparameters help to control how the model learns. n_estimators controls how many individual Decision Trees are used in the ensemble. max_depth controls the maximum depth of the Random Forest. min_samples_split specifies the minimum number of samples needed to split a non-leaf node when training. In tandem, these parameters can help avoid overfitting by limiting the depth of the forest, and how willing the forest is to create new nodes to finely fit data. With a grid search, the optimal parameters, which minimized loss, were found to be: n_estimators = 1000 , max_depth = 15, min_samples_split = 100 . When testing models, a train/test split of 67/33% was used, allowing for the evaluation of the model's performance on unseen data.

To evaluate the Random Forest Regressor, two metrics were used, a coefficient of determination score (R2), and Mean Squared Error (MSE). An R2 score attempts to assess the goodness of fit of a model, where values range from -1 to 1, with values of 1 being perfect. MSE attempts to assess the total error in predictions. This value is mostly arbitrary and depends on the size and magnitude of the data itself, but can be used to compare the performance of different models on a common dataset, as we are doing in this project.

With the hyperparameters of n_estimators = 1000, min_samples_split = 100 , and max_depth = 15, the following results were found:

R2 - Training - 0.299
R2 - Test - 0.217
MSE - Test - 1502.67
Training Time - 776s
Prediction Time - 2.9s

With default hyperparameters of n_estimators = 100, min_samples_split = 2 , and max_depth = None, the following results were found:

R2 - Training - 0.710
R2 - Test - 0.020
MSE - Test - 1881.39
Training Time - 103.8
Testing Time - 1.06

From these results, it is clear how much of an impact finding ideal hyperparameters has in this Random Forest Regressor with regards to testing performance. With default hyperparameters, the model is shown to have a higher training R2 of 0.710, but a much lower testing R2 of 0.020. Compared with the model with custom hyperparameters, having an training R2 of 0.299 and testing R2 of 0.217. For the custom model, the training and testing performance is quite similar, indicating good generalization, compared to the default model with higher training R2, but much lower testing R2. A similar notion can be made with the MSE with test data, where the default model has a MSE of 1881.4, and custom model a MSE of 1502.67. These results indicate that the custom parameters limit overfitting by limiting the depth, and requiring a larger threshold for nodes to be split. Interestingly though, the time required to train and test is much greater for the custom random forest compared to the default random forest.

Although the R2 values in the custom model are far from the ideal value of 1.0, an R2 value of 0.217 still indicates the model was able to learn from the training data, and generalize what it learned to testing data.

Linear Models - Correlation

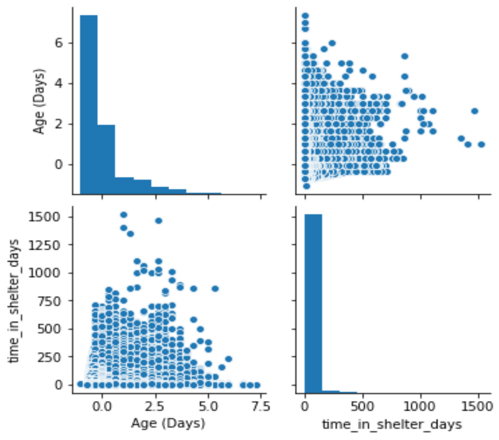

Figure 8. Scatterplot Matrix to View Pair-wise Correlation

Age was the only ordinal variable in our dataset and we had used this feature and the target variable on simple linear models.

We see that the standardized Age (Days) feature has little correlation with the target variable. This is further supported by the correlation matrix below.

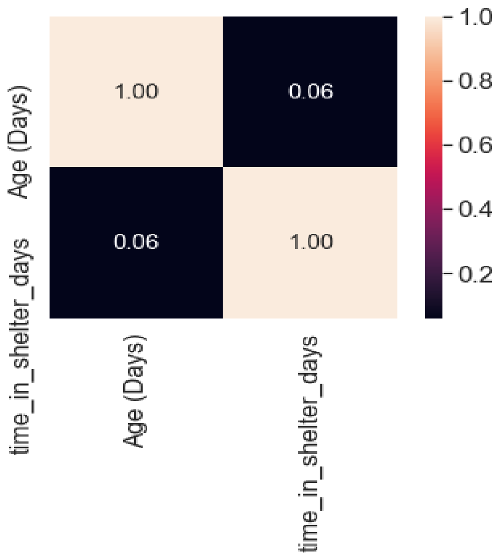

Figure 9. Correlation Matrix

We see that the correlation between Age and the target column is at 0.06. It should be noted that Age is positively skewed- which indicates that the younger animals in the shelter were adopted faster.

Linear Regression

We have performed Linear Regression offered by Sci-Kit Learn Library.

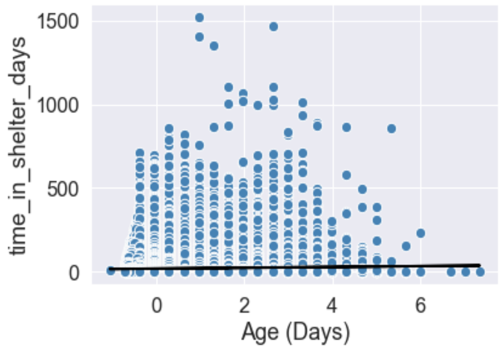

Figure 10. Data Points and Regression Line

The R^2 score on train data was at 0.0033 and test data was at 0.0028. This is very poor performance.

RANSAC Regression

We have applied the Random Sample Consensus (RANSAC) algorithm on the Age feature. RANSAC fits a regression model to a subset of data- the inliers. Later, it selects a random number of samples to be inliers and fit the model. All other data points are tested against the fitted model and those points that fall within a user-given tolerance to the inliers are added. The model is then refitted with all inliers. We then estimate the error of the fitted model versus the inliers. The algorithm is terminated if the performance meets a certain-user defined threshold or if a fixed number of iterations were reached. Else, we again select a random number of samples to be inliers to fit the model and repeat the steps above.

Based on the steps above, we have applied the RANSAC algorithm. We have plotted the inliers and outliers from the fitted RANSAC-linear regression model and below is the result.

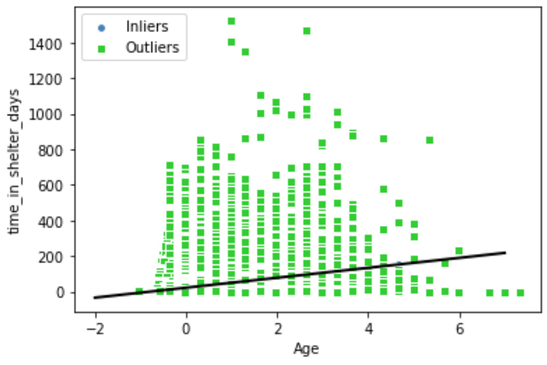

Figure 10. RANSAC Based Model

With little to no correlation, we see that the RANSAC algorithm was not able to detect inlier points. This is coherent with our earlier finding of the correlation between the Age feature and our target variable.

Neural Network Regression

Deep learning methods were also attempted, particularly a Neural Network Regressor. In this model, many layers of individual neurons are used to continuously predict the Time in Shelter. With this model, one-hot encoding was used, rather than label encoding. This is essential, as with Deep Learning, it is essential that the neural network does not attempt to resolve relationships between encoded data that do not actually exist. Before processing and encoding, we had eight feature sets including seven nominal and one ordinal sets, and many more after processing.

We were attempting to design neural networks to predict a continuous real number with an input set that mostly contains categorical variables.

With little to no correlation between our feature set and our target variable- when designing simple neural networks- we faced the challenge of deciding the convenient number of nodes at each hidden layer and a convenient optimizer that would be able to use these features to predict our target variable, a continuous real number from nominal variable inputs.

Designing several possible neural network architectures attempted but none were logically feasible.

Recommendation

The combination of different data preprocessing techniques and different algorithms gave different results. It was observed that with mapping each nominal variable onto an integer with label encoding returned high R2 score, however this could just be a result of overfitting. Since our target variables are continuous real numbers, we believe the regression algorithms formed correlation between the integer labeled ordinal feature set and the target variable. We believe that with further domain knowledge to construct ordinal order within the feature sets, the algorithms will improve in performance and we will have more robust models for predictive analysis. Thus, we recommend forming ordinal scales on the feature sets with domain knowledge.

Conclusion

Applying machine learning techniques to a real world problem was a rewarding experience, even though the performance of the models were low. The results from the Random Forest regressor showed the best results out of all the models employed for this project. As we have learned in class, there is not a “one size fits all” algorithm. Finding an algorithm that best explains a given data set is an iterative process. We have explored some of the most popular methods in regression analysis and feel that in the future we could expand on our work, particularly by developing other and deeper neural networks. This project also taught us the importance of data quality and domain knowledge. It is unknown why some of the outcome dates were before the intake dates, perhaps human error when entering the data. It is also unclear if our grouping of breed and color to reduce cardinality is sound, a veterinarian or veterinarian technician could help us better understand the data. In a general sense, it is also difficult to say whether any sort of model could be constructed to predict time an animal spends in shelter, given the data at hand. There are likely many more factors that go into how quickly an animal is adopted which weren’t given as part of the original dataset, or can’t be represented even in an ideal dataset, such as how the animal appears beyond its breed and color, it’s temperament, and schedules of the animal shelter which may affect when adoptions or transfers are possible. However, given the random forest was able to make sense from the training data and give somewhat educated predictions, it is likely that under more ideal circumstances, and with time, a better model could be constructed.