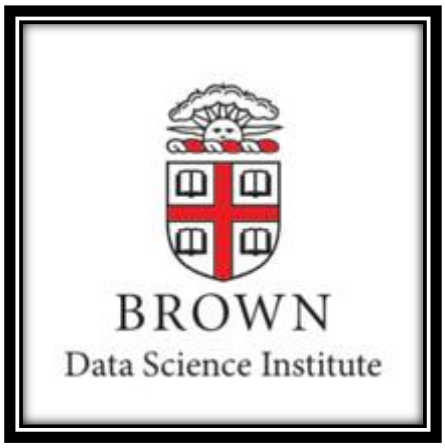

Predicting Shelter Animal Outcomes at Intake Using Machine Learning

Name: Vinayak Mokashi

Course: DATA 1030 (Hands-on Data Science), Fall 2025

Github: Click here

Introduction

Motivation:

Shelters are overwhelmed. More animals come in than get adopted. They're short on staff and vets. Animals are staying longer because they have more serious medical or behavioral issues. All of this means overcrowding, which sometimes leads to outcomes nobody wants and makes it harder to take in new animals who need help [1 ,2].

If we can predict which animals are at risk of poor outcomes right when they arrive, shelters can actually do something about it instead of just reacting when it's too late. They can put limited resources like extra marketing, foster homes, fee waivers, medical care towards the animals who need them most.

Dataset Description:

This project uses two public datasets from the Austin Animal Center: an intakes table [3] (information recorded when an animal arrives) and an outcomes table [4] (information recorded when the animal leaves). The data was obtained from the Austin Animal Center website. Both tables contain a unique AnimalID, and AnimalIDs can repeat because the same animal may re-enter the shelter multiple times. To align with the goal of predicting outcomes *at intake time*, we retain only the most recent visit per AnimalID and create a visit_count feature representing how many total times that animal appeared in the historical records.

We merge the intakes and outcomes tables on AnimalID, and restrict the analysis to cats and dogs. The prediction target is outcome_type with 6 classes. Importantly, after merging, we keep only the predictors that would realistically be available at the moment of intake (to avoid leakage). Missing values occur only in a small subset of categorical variables: missing name is captured via HasName = 0, and missing sex_upon_intake is mapped to an “Unknown” category.

Summary:

- Rows: 135,206
- Features (predictors): 13
- Target: 1 (outcome_type)
- Outcome classes (6): Adoption, Return to Owner, Transfer, Euthanasia, Died, Other
- Time range – 1st October 2013 to 2nd May 2025

For a detailed description of the data, please check here

Previous Work:

Several studies have used the Austin Animal Center dataset to develop machine learning models for predicting shelter animal outcomes. These studies aim to help shelters identify at-risk animals early and allocate resources more effectively. One study using ensemble methods with logistic regression, random forest, and calibrated classifiers achieved a log loss of 0.92 on outcome prediction [5]. Another found that XGBoost showed the highest accuracy at 82%, while random forest achieved 81% accuracy on the same multiclass classification task [6]. A separate analysis focused on binary classification (euthanasia vs. adoption) reported that XGBoost captured 83% of animals at risk of euthanasia correctly [7]. However, these studies used different evaluation metrics than our approach, making direct performance comparisons difficult.

Exploratory Data Analysis (EDA)

We begin by examining the class distribution of the target variable to understand the balance. The outcome distribution is highly imbalanced, with Adoption and Transfer dominating, while outcomes like Died and Other are rare, motivating stratified splits and evaluation metrics that handle imbalance well.

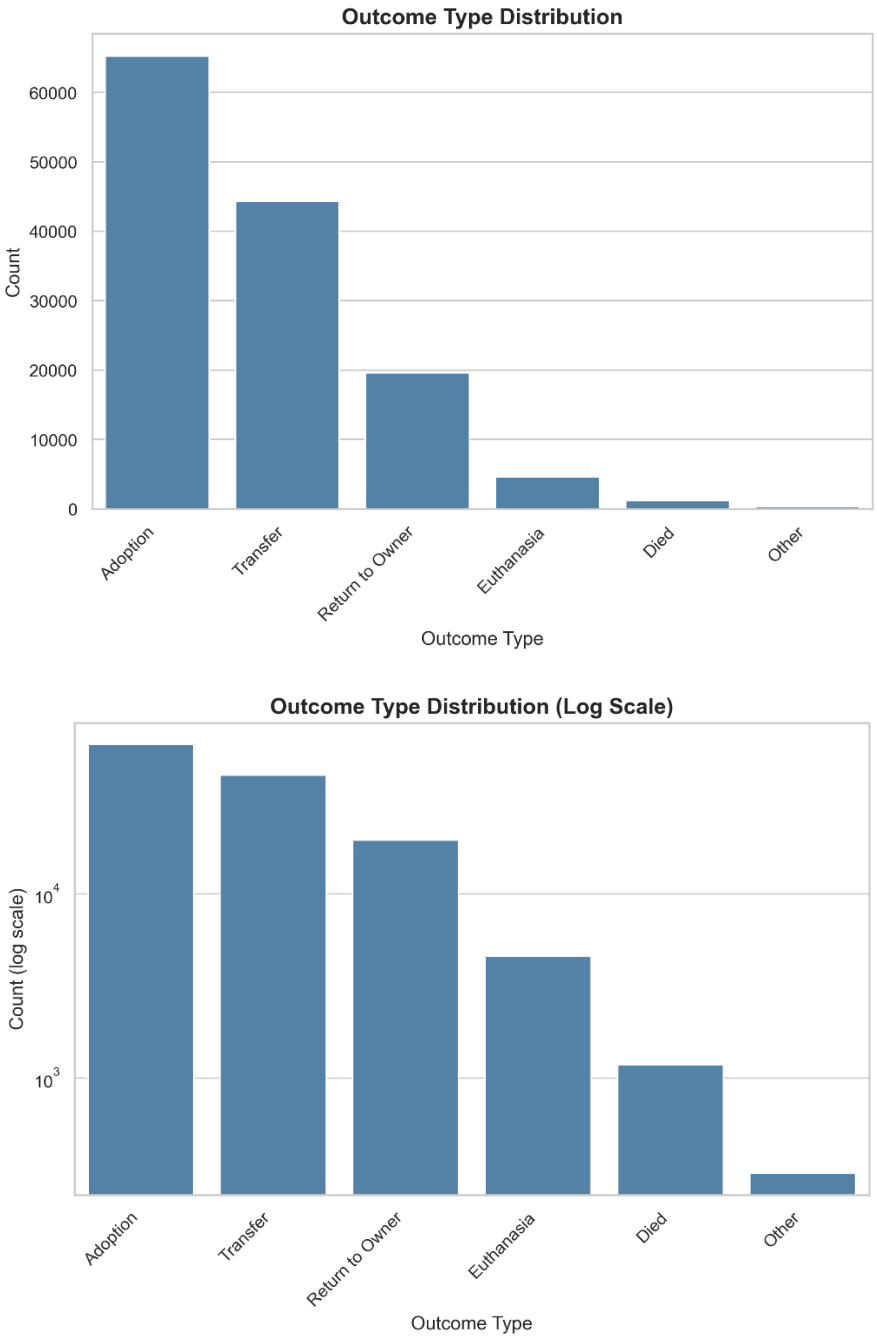

Fig 1 & 2: Outcome Type Distribution (count and log scale)

Next, we explored relationships between outcomes and several features. First, we inspected color vs the outcomes and we suspect that there is a possible color bias in adoption decisions. The normalized outcome proportions vary by color for both cats and dogs. It may carry predictive signal.

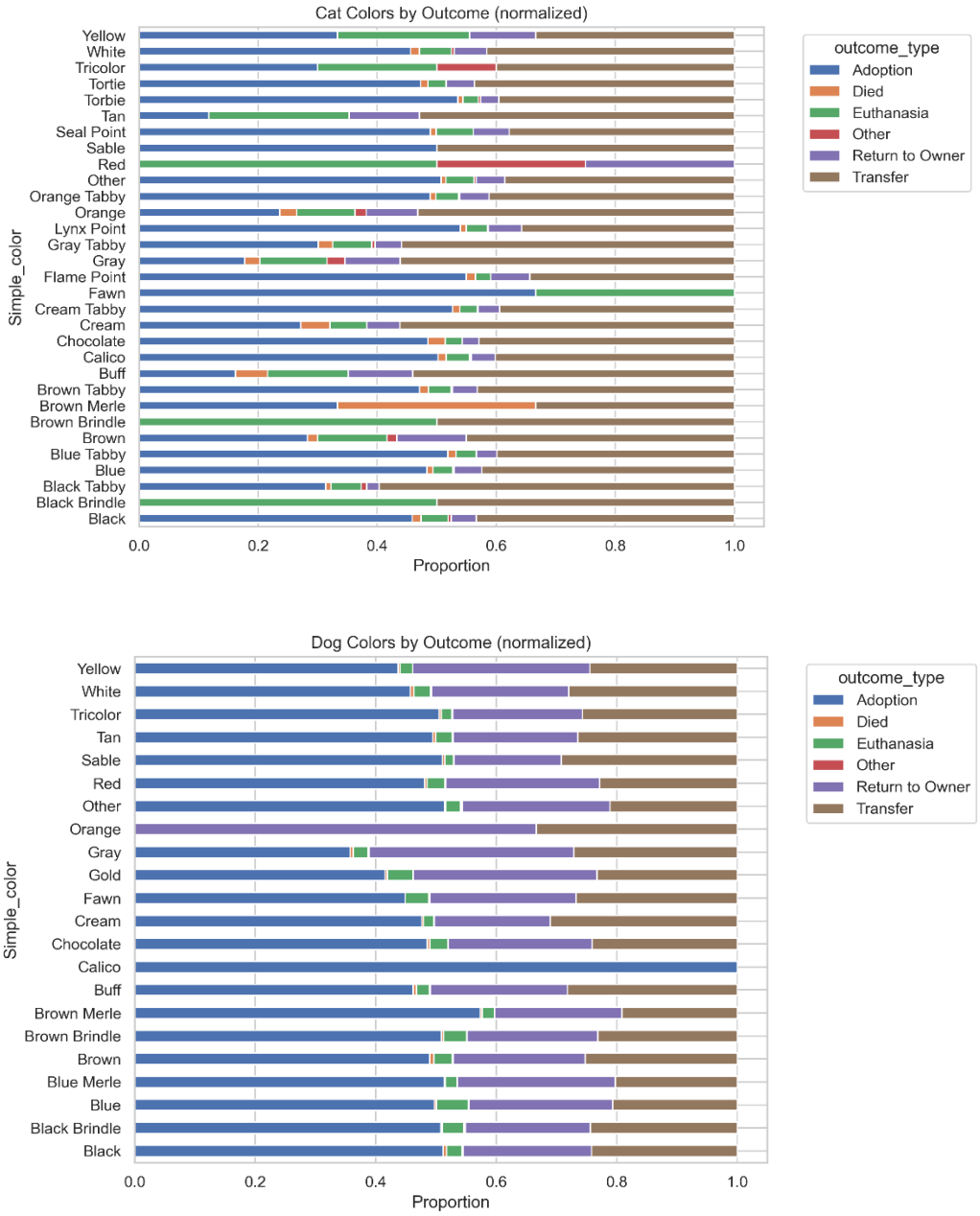

Fig 3 & 4: Outcome by Color (Dogs vs Cats, normalized)

Age shows strong separation: younger animals are substantially more likely to be adopted, while seniors show higher rates of unfavorable outcomes.

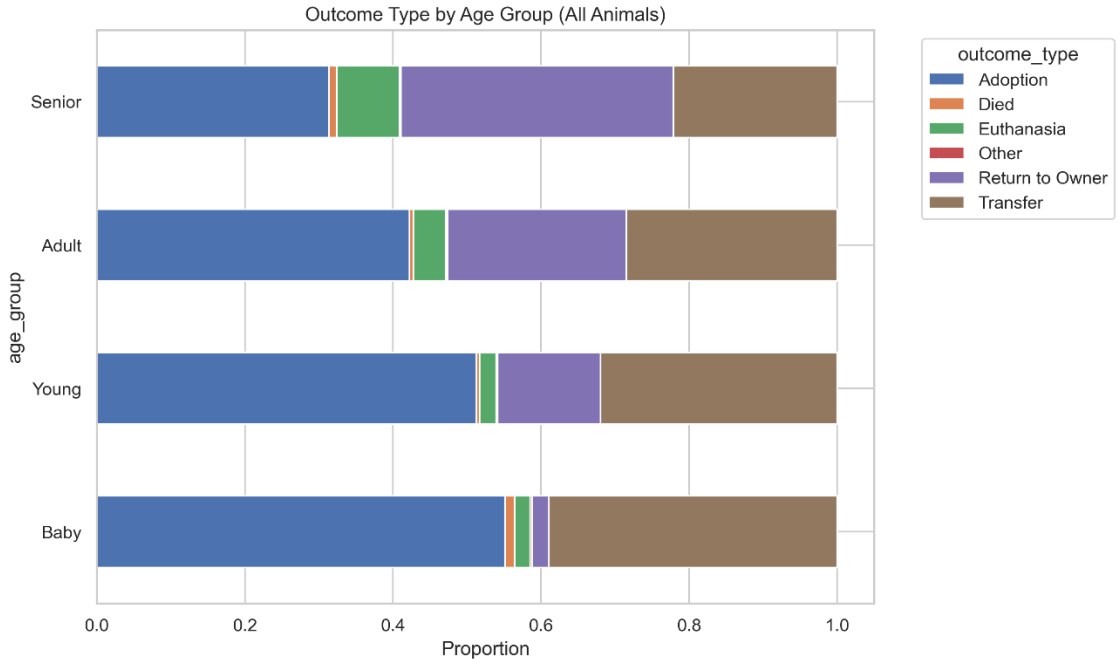

Fig 5: Outcome by Age Group (Cats vs Dogs, normalized)

HasName is also informative: named animals tend to have higher adoption and return-to-owner proportions, suggesting that naming may correlate with familiarity/ownership.

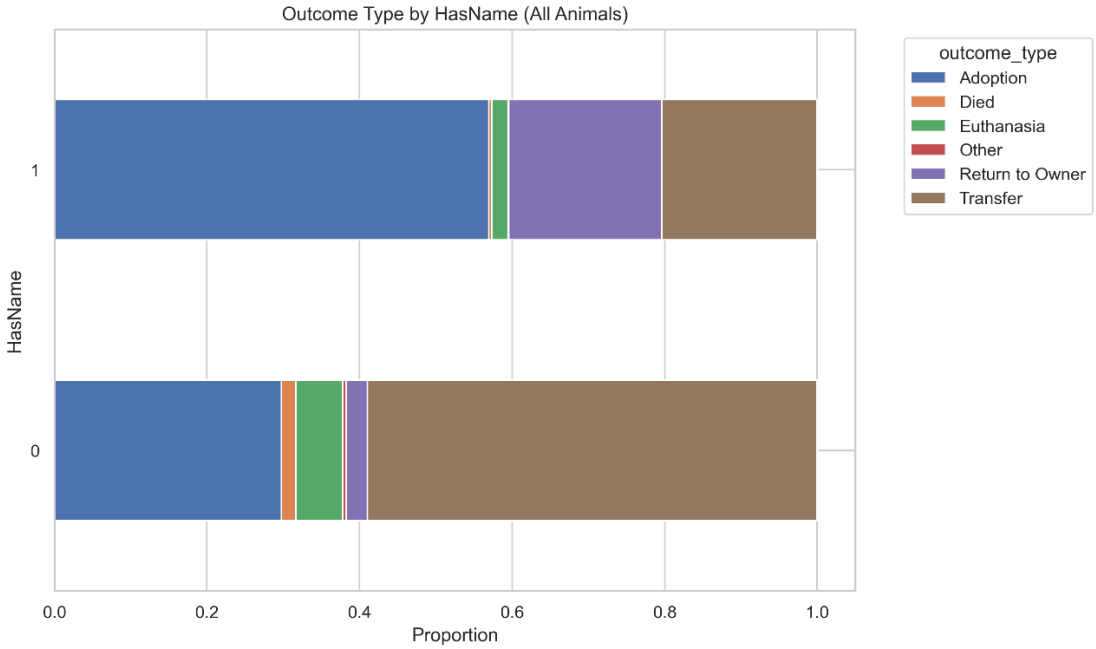

Fig 6: Outcome by HasName (All animals, normalized)

Methods

Splitting Strategy

We use a stratified split on outcome_type to preserve the class proportions across Train/Val/Test = 60/20/20. This ensures rare outcomes appear in all splits and makes comparisons fair. The data is i.i.d. and we avoid K-fold Cross Validation because it becomes computationally expensive with 135K rows and very high-dimensional features after one-hot encoding.

We also run all experiments on a fixed 10% random subset of the full dataset to reduce runtime and memory, while keeping the sample size large enough to retain class diversity. We estimate uncertainty by repeating the full pipeline 5 times with different random seeds. These seeds affect both the data split and (for stochastic models) the training process (e.g., Random Forest bootstrapping/feature subsampling and XGBoost row/column subsampling). We report mean ± std of the test metric across runs.

Data Preprocessing:

We fit the transformers only on the training set, then transform all the three sets.

- Numeric (2): scaled using StandardScaler so values are on a similar range.
- Datetime (3: month, weekday, hour): converted using cyclical encoding (sin/cos) so time wraps correctly (e.g., Dec is close to Jan).
- Categorical (8): encoded using one-hot encoding, creating dummy columns.

We start with 13 input features, and after preprocessing we end up with 950+ features because some columns (like breed/color) have many categories.

Evaluation metric:

We evaluate using Macro-F2 score. This score cares more about recall than precision, which fits our goal: it's worse to miss animals that may need extra help (false negatives) than to flag a few extra animals for support (false positives). We use macro averaging so each

outcome type counts equally, even though some outcomes are much more common than others.

As a simple reference point, we also report a baseline that always predicts the most common class (Adoption) to show how much better our models are than a naive approach.

ML Pipeline:

For each of 5 random seeds:

1. Stratified 60/20/20 split (Train/Val/Test).
2. Preprocess features using train set statistics.
3. Handle class imbalance using class/sample weights derived from Train labels.
4. Hyperparameter search: train models on Train, select hyperparameters with best Validation Macro-F2.
5. Evaluate the selected configuration on the Test set and store Test Macro-F2.

After 5 runs, we:

- report Test Macro-F2 = mean ± std, and
- choose the best model configuration using best average validation Macro-F2 across seeds.

Models and Hyperparameters:

We repeat everything in the above pipeline for the 4 models we use: Logistic Regression (ElasticNet), SVM, Random Forest, and XGBoost classifier algorithm. For each algorithm, several parameters were tuned (Table 1).

Model	Tuned Hyperparameters	Search Space
Logistic Regression (Elastic Net)	C [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000] l1_ratio [0.0, 0.2, 0.4, 0.6, 0.8, 1.0]	8 × 6 = 48
SVM	C [0.001, 0.01, 0.1, 1, 10, 100, 1000] gamma [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000] kernel ['linear', 'rbf']	7 + (7×8) = 63
Random Forest	max_depth [1, 2, 3, 5, 8, 10, 15, 20] max_features ['sqrt', 'log2', 0.1, 0.3, 0.5, 0.7, 0.9, 1.0]	8 × 8 = 64
XGBoost	max_depth [1, 3, 5, 10, 20] reg_alpha [0.0, 0.01, 0.1, 1.0, 10.0, 100.0] reg_lambda [0.0, 0.01, 0.1, 1.0, 10.0, 100.0]	6 × 6 × 5 = 180

Table 1: Tuned parameter values for each machine learning algorithm

Random Forest used n_estimators=500. XGBoost used n_estimators=10000 with early stopping=50 rounds, and fixed learning_rate=0.03, colsample_bytree=0.9, subsample=0.66. For all models, we used balanced class/sample weights, giving higher weight to rare outcome classes.

Results

The Baseline Macro-F2 scores for test and validation sets are 0.1365 for both and there is no variability since we are using a stratified sampling (predicting Adoption all the time). The best model was selected based on the highest average validation scores across all the random sets. We see the best hyperparameter combination was max_depth = 3, reg_alpha = 0 and reg_lambda = 0 for XGBoost (Table 2). The summary of all the models' performance on the validation set can be seen in Table 2 with the improvement over the baseline too. We can also see the summary of the models' performance on the test set in Table 3 and Figure 7 to see the generalization error.

Model to deploy – XGBoost

- Mean Test Macro-F2 is ~17 standard deviations above the baseline Test Macro-F2 (Table 3)
- Validation Macro-F2 is ~42 standard deviations above the baseline Validation Macro-F2 (Table 2)

Model	Best Hyperparameter	Best Average Validation Score	Improvement from Baseline (+)
Logistic Regression (Elastic Net)	C = 0.01 l1_ratio = 0.2	0.343 ± 0.010	0.207
SVM	C = 1 gamma = 0.1 kernel ['linear', 'rbf']	0.381 ± 0.018	0.245
Random Forest	max_depth = 20 max_features = 0.9	0.376 ± 0.006	0.239
XGBoost	max_depth = 3 reg_alpha = 0 reg_lambda = 0	0.387 ± 0.006	0.25

Table 2: Best hyperparameters and validation scores for each model.

Model	Aggregated Test Score	Improvement from Baseline (+)
Logistic Regression (Elastic Net)	0.341 ± 0.022	0.2046
SVM	0.375 ± 0.014	0.2381
Random Forest	0.3723 ± 0.070	0.2358
XGBoost	0.375 ± 0.014	0.2381

Table 3: Test set performance comparison showing improvement over baseline

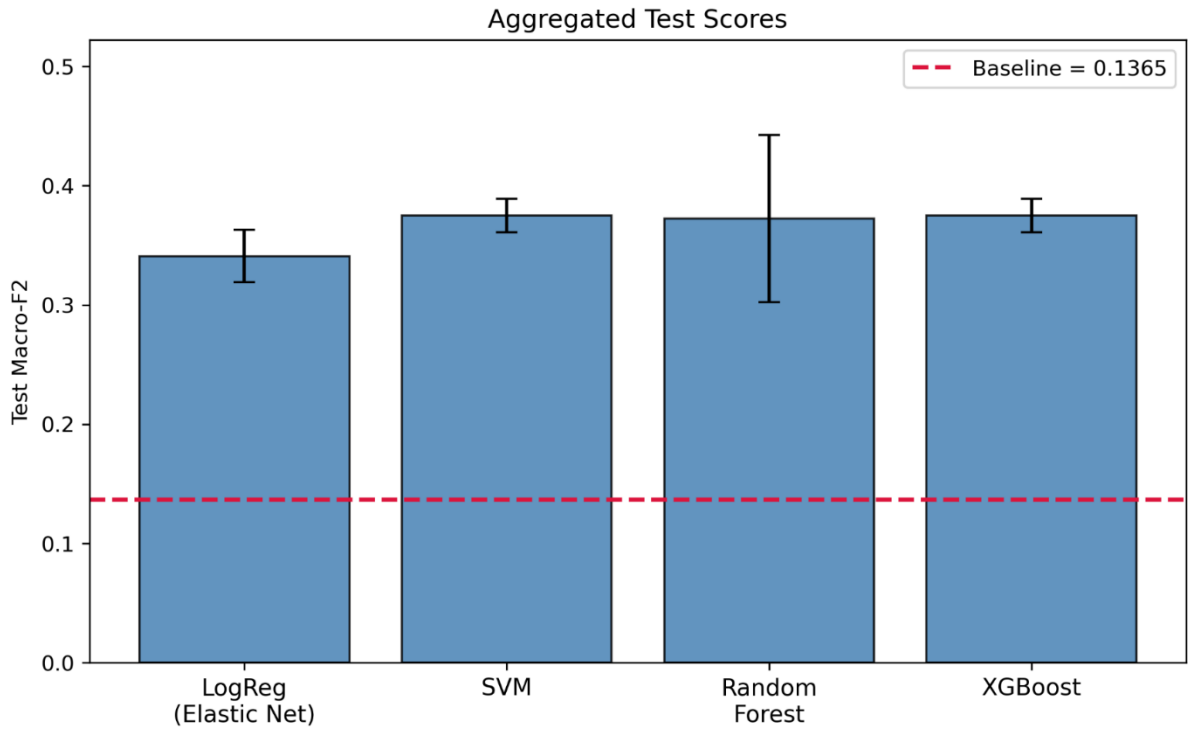

Fig 7: Aggregated test Macro-F2 scores with standard error bars. Dashed line shows baseline (0.1365)

We now inspect our selected XGBoost model. The confusion matrix (Figure 8) reveals the following insights:

Strengths:

- Strong performance on Return to Owner (high recall)
- Good recall for Adoption and Transfer classes

Severe Weaknesses:

- Low or zero recall for Euthanasia, Died, and Other classes (many false negatives)
- The "Other" class is never predicted (model collapses for this class)

Deployment Implications:

- Not deployment-ready for risk flagging
- Only tentatively useful for high-level capacity planning

Global Feature Importance

To understand which features are most influential in our model's predictions, we calculated three different global feature importance metrics: Permutation Feature Importance (Figure 9), Total Gain (Figure 10), and Global SHAP values (Figure 11).

The following features consistently appeared as important across all three ranking methods:

- age_days_intake
- HasName
- intake_condition
- intake_type
- intactness
- intake_hour

These features represent the most reliable predictors in our model, as their importance is confirmed by multiple independent measurement approaches. This suggests that an animal's age, whether it has a name, its condition and intake circumstances, reproductive status, and time of arrival are all critical factors in predicting shelter outcomes.

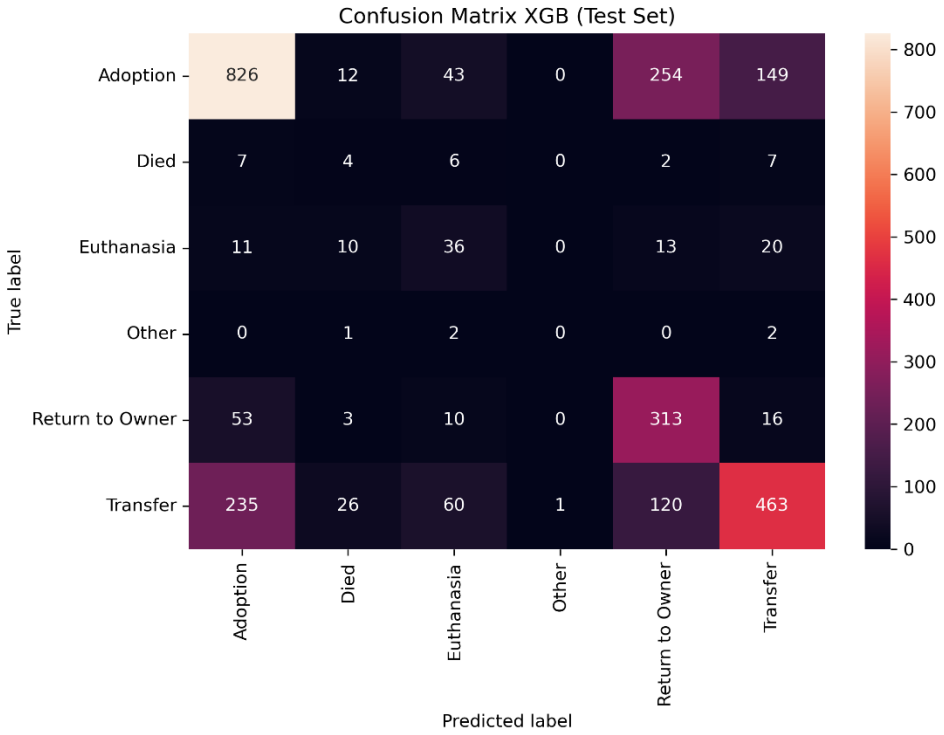

Fig 8: Confusion matrix for XGBoost model on test set

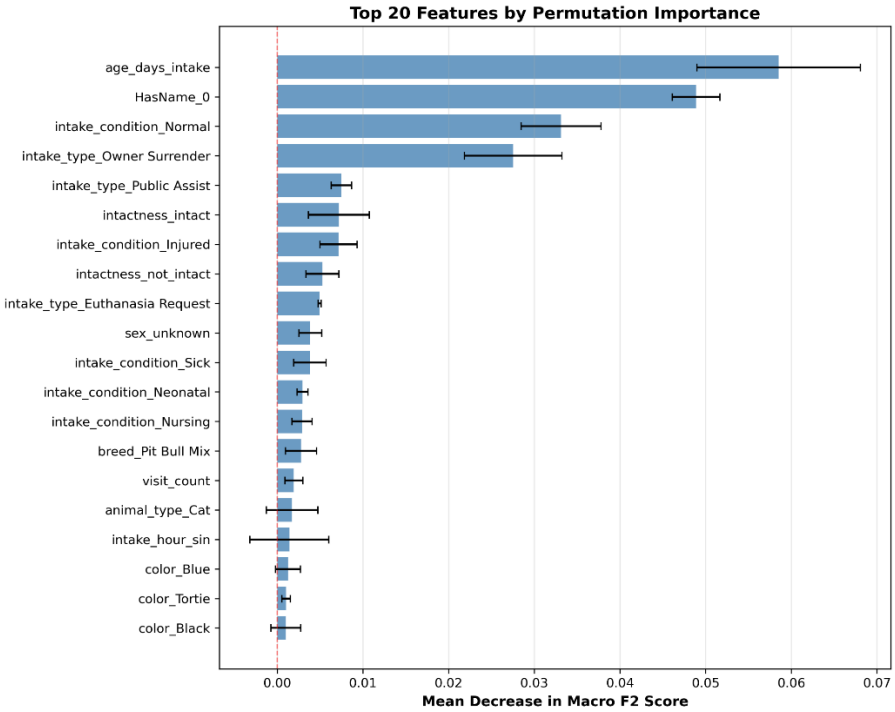

Fig 9: Top 20 features ranked by Permutation Feature Importance

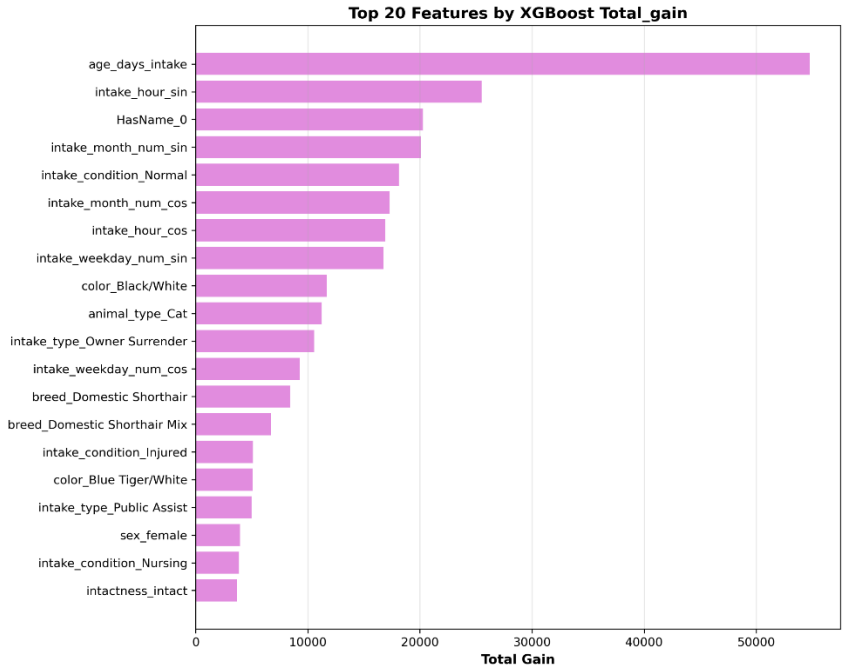

Fig 10: Top 20 features ranked by XGBoost Total Gain

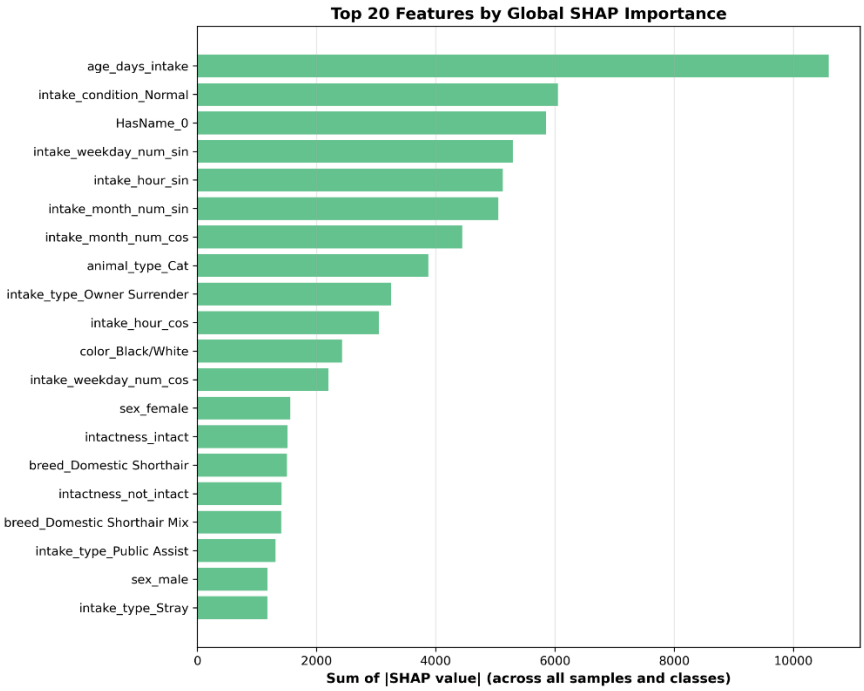

Fig 11: Top 20 features ranked by Global SHAP values

Local Feature Importance - SHAP Force Plots

Next we calculate the local feature importance using SHAP values. It would help us understand how each feature influences the prediction for a particular sample.

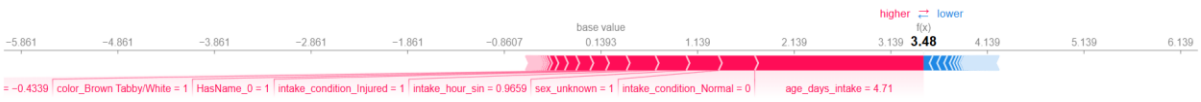

Fig 12: SHAP force plot for Euthanasia prediction (Test Sample 2497, Confidence 0.982).

This force plot shows how different features pushed the model toward predicting Euthanasia for this specific animal. The base value (0.1393) represents the average model output. Red arrows indicate features that increase the Euthanasia prediction, while blue arrows push against it. For this sample, factors like not having a name (HasName_0 = 1), being injured at intake (intake_condition_Injured = 1), and having unknown sex strongly pushed toward Euthanasia, while younger age (age_days_intake = 4.71) pushed against it. The final prediction value of 3.48 shows high confidence for Euthanasia.

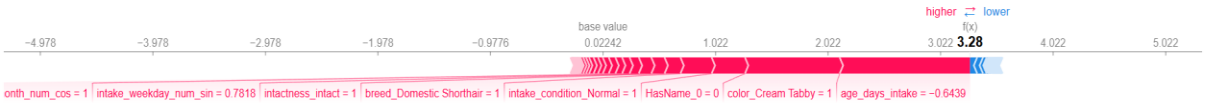

Fig 13: SHAP force plot for Adoption prediction (Test Sample 2658, Confidence 0.939).

This plot shows the model correctly predicted Adoption with high confidence (3.28). Key factors pushing toward Adoption include: having a name (HasName_0 = 0), being in normal condition at intake (intake_condition_Normal = 1), being intact (intactness_intact = 1), and being a Domestic Shorthair. The younger age (age_days_intake = -0.6439, meaning very young) also contributed positively to the Adoption prediction.

The most important features were age_days_intake, HasName, intake_condition, intake_type, intactness, and intake_hour. This aligns with shelter realities. Younger animals with names (suggesting prior ownership) in good health are more adoptable. Surprisingly, temporal features like intake_hour ranked highly, suggesting operational factors matter more than expected, while breed and color were less important than conventional wisdom suggests. The SHAP force plots revealed why the model struggles with minority classes: it has learned that most animals get adopted or transferred but fails to capture the specific factor combinations leading to euthanasia or death, precisely what shelters need to predict for intervention.

Outlook

The main weak spot of our modeling approach is the severe class imbalance, where the model fails to predict minority classes (Euthanasia, Died, Other) that are most critical for intervention. To improve this model, we could increase class weights or apply SMOTE to balance the training data, or simplify the problem to binary classification (positive outcome vs. at-risk) to better identify animals needing intervention. Additionally, using the whole dataset instead of our sampled 10% would boost predictive power and reduce variance. For improved interpretability and accuracy, richer feature engineering could help, grouping color and breed into behaviorally meaningful clusters rather than treating each as separate categories, and adding context features like current shelter population size and demographic breakdowns by animal type and age group. We could also reframe the task entirely to predict length of stay or time-to-outcome, allowing shelters to allocate resources for animals likely to stay long. Finally, we should audit predictions by breed, color, age, and intake type to detect and reduce potential discrimination, ensuring the model serves all animals fairly.

References

[1] ASPCA. "U.S. Animal Shelter Statistics." 2024. https://www.aspca.org/helping-shelters-people-pets/us-animal-shelter-statistics

[2] NBC News. "Animal shelters are crowded as high costs squeeze pet owners." July 28, 2025. https://www.nbcnews.com/business/economy/animal-shelters-full-pets-expensive-inflation-rcna221043

[3] Austin Animal Center Intakes (10/01/2013 to 05/05/2025)

https://data.austintexas.gov/Health-and-Community-Services/Austin-Animal-Center-Intakes-10-01-2013-to-05-05-2/wter-evkm/about_data

[4] Austin Animal Center Outcomes (10/01/2013 to 05/05/2025)

https://data.austintexas.gov/Health-and-Community-Services/Austin-Animal-Center-Outcomes-10-01-2013-to-05-05-/9t4d-g238/about_data

[5] Katy-katy. "Shelter-Animal-Outcomes-Machine-Learning-Python." GitHub repository. https://github.com/Katy-katy/Shelter-Animal-Outcomes-Machine-Learning-Python

[6] CjMullins87. "Animal-Shelter-Outcomes: Predicting animal shelter outcomes using classification models." GitHub repository. https://github.com/CjMullins87/Animal-Shelter-Outcomes

[7] Huang, Y. "Animal Adoption-How Data Science Can be Used to Help Animals in Shelter?" Medium, January 7, 2022. https://e-82849.medium.com/animal-adoption-how-data-science-can-be-used-to-help-animals-in-shelter-30b980db7403