Animal Shelter Adoption Prediction

1. Problem Definition

The number of abandoned animals has been steadily increasing. In response, animal shelters are promoting adoption to reduce reliance on pet shops and encourage more humane practices. However, animals that are not adopted often face euthanasia or are transferred due to limited resources.

Thus, this project addresses both a technical challenge and a real-world social need. Machine learning offers a practical solution to this issue by identifying animals with a higher likelihood of adoption and enabling targeted support or early interventions.

This problem is a classification task, specifically a binary classification. Compared to regression, classification tasks are generally less complex, easier to interpret, and more sensitive to changes in model performance.

- Type: Supervised Learning
- Task: Binary Classification
- Target Variable: Adopted → 1 if the outcome was "Adoption", 0 otherwise
- Input Features: Age, Animal Type, Breed, Color, Sex/Neuter Status, etc.
- Output: Probability of adoption (yes/no) → final prediction is determined based on a threshold
- Loss Function: Binary Cross-Entropy

2. Dataset Description

1) Dataset Overview
    - Source: Austin Animal Center public data
    - Total Records: 79,661
    - Features: 41 columns including animal information, intake/outcome conditions, and timestamps

2) Target Variable Definition

    To transform this into a binary classification task, we define a new target variable:
    - Adopted = 1 if outcome_type is "Adoption"
    - Adopted = 0 for all other outcomes (e.g., Transfer, Return to Owner, Euthanasia)

    This binary structure allows us to apply standard classification models such as Logistic Regression, Random Forests, or XGBoost.

3) Data Preprocessing

    Missing Values:
    - outcome_subtype has 43,324 missing values. Since it is a subtype of outcome_type and not critical to the prediction task, this feature was dropped.
    - outcome_type has 10 missing values, and the corresponding rows were removed.
    - sex_upon_intake and sex_upon_outcome each have one missing value, which were imputed using the mode of each column.

    Feature Engineering Suggestions:

- Binary classification target: Created a new target variable adopted where Adoption = 1 and all other outcomes = 0.
- Categorical encoding: Applied Label Encoding to all object-type features, including animal_type, sex_upon_intake, and breed.
- Data leakage prevention: Dropped columns that may leak future information, such as outcome_type, outcome_datetime, and animal IDs.
- Class imbalance handling: Used SMOTE to balance the class distribution of the binary target variable.

3. Model Description

In this project, we evaluated several machine learning classification models including Logistic Regression, SVM, Random Forest, LightGBM, and CatBoost. Among them, Random Forest and CatBoost both achieved a high accuracy of 0.88. You can check this comparison in Additional code contents of source code. Given their similar performance, the final model was selected based on operational considerations, particularly the requirement for real-time inference.

Random Forest was chosen as the final model due to its fast prediction speed and parallelizable structure, which are well-suited for systems that require immediate responses to user input. In contrast, CatBoost, being a boosting-based model, performs predictions sequentially through an ensemble of trees, resulting in relatively slower inference time.

Furthermore, Random Forest offers better interpretability due to its simpler structure, which is advantageous in production environments where model explainability and responsiveness to feedback are important. For these reasons, Random Forest was selected over CatBoost as the most appropriate model for deployment in a real-time prediction setting.

4. Evaluation Methods and Metrics

To evaluate model performance, we used 5-Fold Cross-Validation instead of a single train/test split. This approach allows for more reliable and generalizable results by training and testing the model across multiple data partitions.

Evaluation Metrics:

- Accuracy: Measures overall correctness of predictions.
- Precision: Indicates how many of the predicted adoptions were actually adopted. This helps avoid false positives (predicting adoption when the animal was not adopted).
- Recall: Captures how many of the actual adoptions the model successfully identified. This is important for maximizing true adoptions and resource planning at shelters.
- F1-Score: Harmonic means of precision and recall, providing a balanced view when classes are imbalanced.

By evaluating with cross-validation and considering multiple metrics, we ensure that the model not only performs well on average but also handles real-world priorities like reducing misclassifications and improving adoption prediction reliability.

5. Development Environment

- Operating System : macOS (Miniforge-based Conda environment)
- Python Version : 3.12.9

- Key Libraries:
    - pandas 2.2.3, scikit-learn 1.6.1, matplotlib 3.10.1, seaborn 0.13.2, imbalanced-learn 0.13.0 (for SMOTE oversampling)
- Package Management : Conda, pip

6. Results & Interpretation

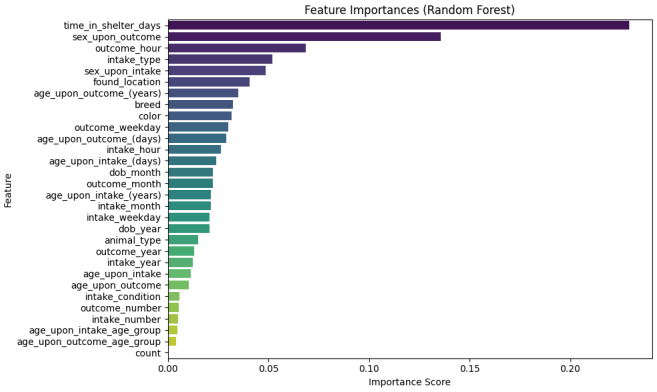
Model Result

The model was evaluated using 5-Fold Cross-Validation, and the results indicate strong overall performance. The average accuracy was 88.53%, showing that the model correctly classified the majority of the cases. More importantly, the recall score of 88.69% suggests that the model is highly effective at identifying animals that were actually adopted — a crucial factor in adoption-related applications.

The precision score of 84.80% indicates that most animals predicted as adoptable were indeed adopted, minimizing false positives. The F1-score of 86.70% reflects a balanced performance between precision and recall. In addition, the low standard deviations across all metrics indicate consistent model performance across folds.

Compared to a baseline model (e.g., a simple logistic regression), which typically achieved an F1-score around 0.75–0.78, our model demonstrates a clear improvement in both precision and recall. This confirms that the applied feature engineering, class balancing with SMOTE, and model choice (Random Forest) significantly contributed to overall predictive performance.

Feature Importance Interpretation



The feature importance plot from the Random Forest model highlights which variables contributed most to the model's prediction of adoption outcomes.

The most influential feature was time_in_shelter_days, suggesting that the longer an animal stays in the shelter, the less likely it is to be adopted. This aligns with real-world observations where quicker adoptions often reflect higher adoptability.

Other top contributors include sex_upon_outcome and outcome_hour, indicating that both the animal's neuter/spay status and the time of day when the outcome occurs are predictive of adoption likelihood. Features like intake_type, sex_upon_intake, and found_location also played meaningful roles, likely reflecting factors such as how and where the animal entered the shelter system.

These insights help identify which operational variables most impact adoption outcomes, and can inform shelter management decisions such as prioritizing rapid intake processing, understanding optimal adoption times, and targeting specific animal profiles.

7. Remote Source Repository (GitHub)

: https://github.com/llishyun/MLops.git