Animal State Prediction

-ANAND

How to run

1. Running the whole Jupyter notebook will produce the predictions for all the different models.

   Animal_Adoption_Prediction_challenge.ipynb

2. The second file contains the EDA performed to identify the relationships between the features and various plots.

   Animal_Adoption_EDA.ipynb

3. The cleaned and engineered data is included in the file

   COMBINED_feature_engineered_partially_cleaned.csv

The notebooks are well documented with explanations behind steps taken.

Approach 1: Animal Adoption Prediction Challenge (Tree based classifiers)

1. **Removed** columns that were **highly correlated** with each other. (age_upon_intake_years and age_upon_intake_days etc. had same info)
2. **Removed** columns that had **already been split** into relevant columns (intake_datetime, date_of_birth etc.)
3. **Feature engineering** to add columns that may contain relevant feature (animal_neutered, animal is mix, sex, weekday when outcome)
4. **Feature Engineering** to **simplify columns/features** that had too many categories (breed, color)

5. **Converted** categorical animal_type to **One Hot Vector.**
6. Used 14 features as training features for various Tree based algorithms and an Artificial Neural Network(ANN).

Models:

1. Used **Decision trees** with GridSearch Optimized parameters and cross validation for overfit checking. – **F1 Score of 47.67%**
2. Used **Random forest ensemble** to reduce variance and get a more stable prediction. - **F1 score of 49.99%**
3. Used **XGBoost** gradient boosting classifier to get better predictions with custom parameters(not cv optimised). – **F1 score of 49.54%**

Random Forest was the best tree based model by a narrow margin.

Approach 2: Animal Adoption Prediction Challenge (Artificial Neural Network)

- One hot encoded Target variable.
- Simple neural network with 1 FC layer with 27 units, relu activation.
- Adam Optimizer and categorical_crossentropy loss for multiclass classification.
- SoftMax final activation for classification.
- Using keras, trained for 20 epochs with a batch size of 32.
- **Predictions give F1 score of 46.35.**
- **Model accuracy during learning very slowly goes above 50, not even overfitting.**
- **CONCLUSION : Better feature engineering required as ANN can’t even overfit/ find structure in data even with GPU (1000 epochs).**

Tools

- Python ML stack, Jupyter notebook, Keras, excel etc…