Predicting the Outcome of Shelter Animals

Ceren Akkalyoncu Aina Centelles Irem Gokcek
Karin Lintzen Harshul Shukla

March 29, 2019

Abstract

The goal of this paper is to explore the effectiveness of predicting the outcome of cats and dogs in animal shelters. The data originates from an animal center in Austin, Texas from 2013 to 2016. Several machine learning models were implemented for this with Multinomial Logistic Regression being the best predictor. Accuracy was moderately low and was increased with the removal of animals that were euthanized or died in the shelter

1 Introduction

Animal Shelters are a place where abandoned or lost animals, mostly cats and dogs, are kept and rehabilitated. In particular, the Austin Animal Center is the largest no-kill center in the United States of America, providing shelter to more than 16000 animals each year. [1]. Their records are available from 2013 to 2016. Using this data set containing the information of 26.700 cats and dogs, a machine learning algorithm was designed to predict the outcome of said animals. By trying several models, the goal was to predict whether an animal was going to be adopted, transferred to another facility, euthanised, returned to their owners or die. Several factors in the data set were thought to influencial such as the animal's breed and age while some other factors, such as how long the animal had been in the shelter, were not included. The data source can be located at `https://www.kaggle.com/c/shelter-animal-outcomes/data`. The machine learning models that were considered included: Naive Bayes, Multinomial Logistic Regression, K Nearest Neighbors, Support Vector Machines, Random Forest and Multilayer Perceptron.

2 Data Wrangling and preparation

2.1 Data Description

The Kaggle dataset has a training and testing .csv file. Unfortunately, the testing data does not actually contain the target column because the data is meant for a competition, so predictions are supposed to be uploaded to Kaggle and evaluated by a black box. Due to this, the training file was used for training and testing. The dataset consists of 26,700 rows where each row is an animal's outcome and traits about them. This information is recorded upon the animal's exit of the facility. All animals in the data were are cats or dogs. The columns include the AnimalID, Name, Date of Exit, Outcome Type, Outcome Subtype, Animal Type, Sex and Reproductive Capability, Age, Breed, and Color. The target column is the OutcomeType.

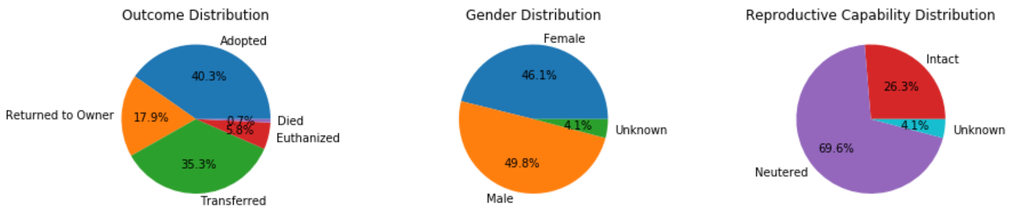
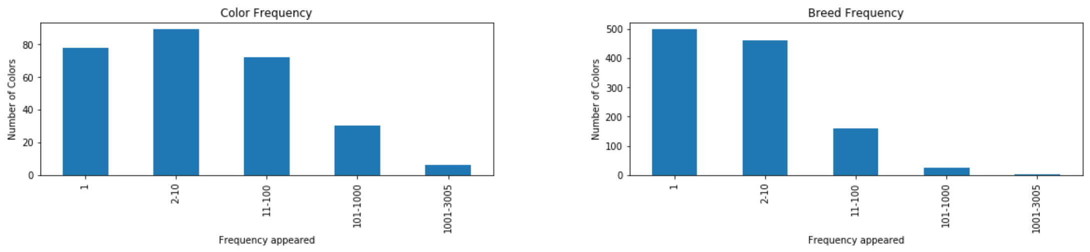


Figure 1: Distribution of the dataset

One can see here that only about 40% of the animals that leave the shelter do so by adoption. Animals that are marked as "Returned to owner" are animals that have taken temporary residence at a shelter after escaping or getting lost. After their owner is found they are returned home. Animals that are "transferred" are done so to avoid euthanasia. Since this dataset is from the South where they are much more open to euthanasia and have lower neutering rates, they transfer animals to more progressive shelters in the West and Northeast where there is a relative shortage of animals and better medical care[1]. So while it is better than euthanasia, a transfer is generally a negative outcome. The gender distribution is close to equal while the reproductive status leans heavily towards neutered animals.



Color and breed frequencies

Color and breed have a similar distribution. There are a lot of colors and breeds that show up just once or very seldom. This is due to the enormous range of unique animal breeds and niche names due to interbreeding. The animals' fur color also has a wide range of color descriptions that are specific and sometimes unheard of at all. This is a bit of a cause for alarm because it will be difficult for the models to make sense of an attribute that only shows up once or twice in the dataset. Furthermore, if this was an online model it would be difficult to understand and interpret new breeds and colors.

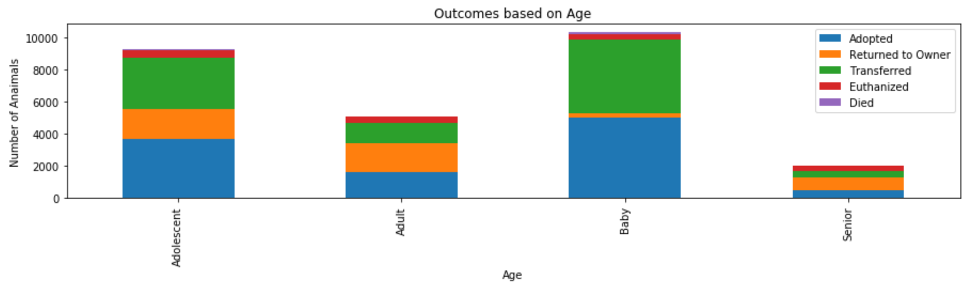
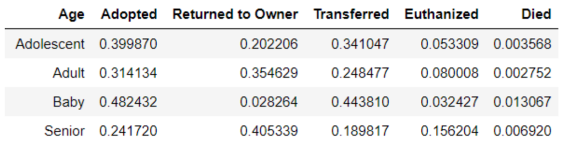


Figure 2: Outcomes based on age



Age	Adopted	Returned to Owner	Transferred	Euthanized	Died
Adolescent	0.399870	0.202206	0.341047	0.053309	0.003568
Adult	0.314134	0.354629	0.248477	0.080008	0.002752
Baby	0.482432	0.028264	0.443810	0.032427	0.013067
Senior	0.241720	0.405339	0.189817	0.156204	0.006920

Figure 3: Distribution of outcomes

One can see here that the distribution of outcome changes significantly depending on the age of the animal. In particular, being younger always has a positive effect on being adopted but also makes the animal more likely to be transferred (most likely due to new litters of animals during spring[2]). Senior animals are most likely to be euthanized since they have low adoption value and suffer from health issues that are too expensive to treat or too terrible to endure.

2.2 Data Cleaning

The dataset was greatly altered to make it more effective for the models. First, the columns *Name*, *Outcome Subtype*, *Animal ID*, and *DateTime* were dropped. The Name, ID, and DateTime were considered features that would have low predictive power because they don't necessarily have value in terms of adoption. Outcome subtype was excluded because it is extra information about the target column and would have likely given the solution away. Furthermore, it makes no sense to use a description of the result while trying to predict the outcome.

Next, the age column was transformed to be categorical and discrete instead of numerical. The brackets that were used for categorization were less than 26 weeks, between 26 and 110 weeks, between 110 weeks and 390 weeks, and older than 390 weeks with the labels `baby`, `adolescent`, `adult`, and `senior` respectively. There are slight variations between dogs and cats and even among different breeds of the same animal, but this was a valid general scheme. Breed was a feature that had some additional difficulties because of the staggering number of unique values. The best way to deal with this is to create two new columns for each

[1] `https://wpde.com/news/local/southern-shelters-sending-animals-north`
[2] `https://wpde.com/news/local/southern-shelters-sending-animals-north`

called `Breed_1` and `Breed_2`. Each row has value that is either just some descriptor A, or a combination of two descriptors A and B separated by a backslash. In the former case, the old value is simply inserted into both new columns while in the latter case one would need to first tokenize the values on each side of backslash and place them in the new columns in alphabetical order. This was done so that rows with the value A/B would be identical to rows with the value B/A. Additionally, a new column `isPurebred` was created where the value was a 0 if the breed value contained the word "mix" or had two different breeds separated by a backslash and a 1 otherwise. Color was transformed in an identical fashion except for the Purebred column.

Lastly, all rows with the target value "return to owner" or "died" were removed. This was due to the fact that both were not reasonable to predict. Animals died at a mostly uniform rate across the different age groups and none of the other features had any correlation with health aside from noise in the data. In the case of return to owner, these rows are wholly irrelevant to the purpose of this exploration. Animals that were returned to owner are only recorded because they were lost and needed a place to stay while their owner was contacted or searched for. They are not actual candidates for adoption, transfer and so forth and therefore have nothing to do with what attributes make an ideal candidate for adoption.

2.3 One Hot Encoding

As most of the features were strings, they had to be converted into integer values, so that the models could process the data. One hot encoding was used for all features except for the output, to prevent numerical value wise comparison between features. The output "OutcomeType" was encoded using label encoder.

3 Experiments and Models

First, the dataset was split into two different portions. The first was for training purposes (80% of the instances) and the other was strictly for testing (20% of the instances). Then, the training data was split again into 5 folds and used K-folds cross-validation. The data was split into 5 equal sets (16% of the total each) where 4 of the sets were used to train and the last used to validate. These sets were rotated such that each acted as the validation set once. After the hyperparameters had been tuned, the model was trained on the entire training set and evaluated with the testing set. The python library *scikit-learn* was used heavily to implement the models and the GridSearchCV package from sklearn was used to find the optimal hyperparameters [5].

3.1 Naïve Bayes

3.1.1 Description

The naïve Bayes classifier [7] is based upon the principle of Maximum A Posteriori (MAP). The naïve classifier assumes that all features are independent and conditional on the class. It's called naïve because the assumption is usually untrue, but the resulting classifier nevertheless works well in many cases [8]. The formula for the Naïve Bayes is as follows, where for i class (Y), given that k attributes (X) is:

$$p(Y = y_i|X_1 = u_1, X_2, ., X_k = u_k) = \frac{p(Y = y_i)\prod_{i=1}^{k} p(X_i|Y = y_k)}{p(X_1 = u_1, X_2 = u_2, ..., X_k = u_k)}$$

The algorithm classifies each of the inputs given the following formula:

$$Y_{predicted} = argmax_v(p(Y = y_v|X_1, X_2, ..., X_k))$$

Because the input data for the Naive Bayes is a binary data frame. Therefore we use the Bernoulli Naive Bayes [9].

3.1.2 Results

With the naïve Bayes classifier, A training accuracy of 0.550 was obtained, and a test accuracy of 0.541. There is no indication of overfitting in the naïve Bayes classifier. The accuracies differ between the different classes, so is the accuracy of the adoption class 0.67 and the accuracy of the died class 0.00. Figure 4 shows the output of the model.

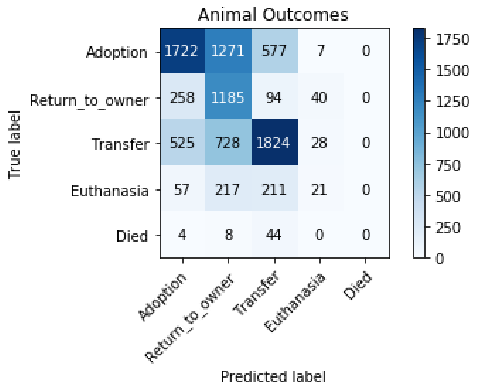


Figure 4: The Confusion matrix of the Naïve Bayes

3.2 Multinomial Logistic Regression

3.2.1 Description

Multinomial logistic regression is a modified version of logistic regression that can handle multiclass problems. It also uses maximum likelihood estimation to evaluate the probability for the different categories. Out of the different solver options sklearn provided, “newton-cg” was used. Newton’s method is similar to a gradient descent but provides a better quadratic function minimization using the Hessian matrix [10].

3.2.2 Results

In multinomial logistic regression, a training accuracy of 0.634 was obtained, and test accuracy of 0.638, indicating that there is no overfitting in the model. As can be observed from the confusion matrix in Figure 5, the model was relatively successful in predicting the "Adoption" and "Transfer" outcome, but it did not produce any reliable prediction for the other classes.

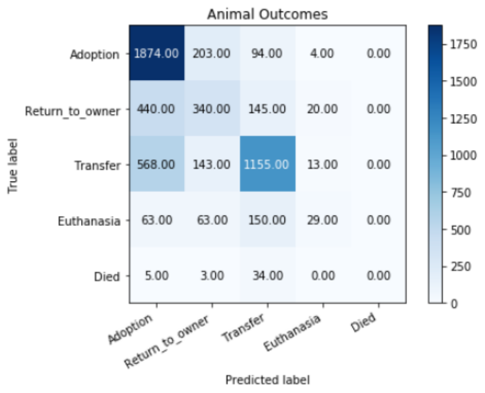


Figure 5: The Confusion matrix of the Multinomial Logistic Regression

3.3 K-Nearest Neighbors

3.3.1 Description

The k-nearest-neighbor (KNN) algorithm is a method used for classification and regression. It classifies how similar an instance is to another. It assigns weights to the contributions of the neighbors where the nearer neighbors contribute more than the distant instances. The number of nearest neighbors, K can be determined [11]. It is the core deciding factor. However, to avoid overfitting, it is better to choose K as not too small. There are different ways of calculating the distance between the neighbors, such as:

- Euclidean: $\sqrt{(}sum((x-y)^2)$
- Manhattan: $sum(|x-y|)$
- Jaccard: $J(A,B) = \frac{|A \cap B|}{|A|+|B|-|A \cup B|}$

3.3.2 Results

First, k=1 was tried and increased until the accuracy was relatively better. The best results were obtained when k=6. The best distance algorithm was Jaccard with training accuracy 0.611 and test accuracy 0.612. The training accuracies obtained from Euclidean and Manhattan were around 0.60 but their test accuracies were lower. That is why a different distance algorithm was considered, like Jaccard. Like other models implemented, KNN could not successfully predict the outcomes "Return to owner" and "Transfer" which was expected. The results were not expected to be higher than the other models since KNN is a relatively basic model.

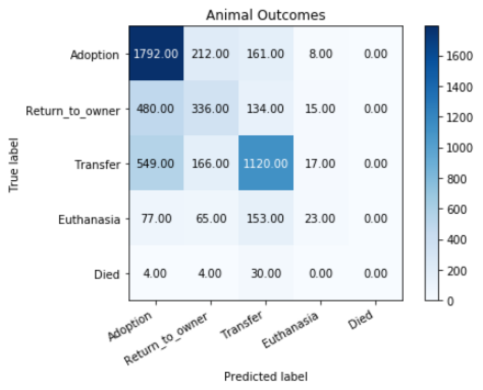


Figure 6: The Confusion matrix of the K-Nearest Neighbors

3.4 Support Vector Machines

3.4.1 Description

Support Vector Machines (SVM) is a supervised machine learning algorithm used for regression and classification problems. It works by plotting the instances as points in space and then finding the hyperplane (or set of hyperplanes) that maximizes the distance between the decision boundary and the nearest points (called the Support Vectors). It works well with linearly separable data as well as non-linearly separable data using the *kernel trick* by plotting in a higher dimensional space where separation is thought to be easier [6] . When using SVM, several hyperparameters are to be chosen. On the one hand, three kernel functions were considered:

- Linear Kernel: $k(a, b) = a^\top b$
- Polynomial Kernel: $k(a, b) = (a^\top b + 1)^d$
- RBF Kernel: $k(a, b) = e^{-\gamma||a-b||}$

On the other hand, several values for $\gamma \in [10^{-3}, 10^{-2}, \ldots, 10^2, 10^3]$ and $C \in [10^{-1}, 1, 10, \ldots, 10^5]$ were considered. This was done with a grid search using *Scikit-learn* and resulted in the optimal parameters to be: $\gamma = 0.1, C = 10$.

3.4.2 Results

With these parameters, the resulting training accuracy was 0.7142 and the test accuracy 0.6233. The classifier did best at predicting the adopted animals and worst at predicting the ones that died, it did not classify any instance as died, as one can see in the following confusion matrix.

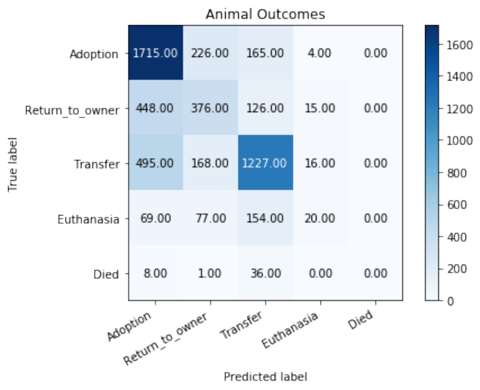


Figure 7: The confusion matrix of the Support Vector Machines

3.5 Random Forest

3.5.1 Description

The random forest algorithm could actually better be described as the random decision tree forest. It creates many decision trees that are trained with the bagging method and then combined to create one final decision tree. This leads to significantly less overfitting and is the reason we that a pure decision tree was not considered. The specific Random Forest algorithm employed by the scikit-learn python package is called CART analysis which was introduced by Leo Breiman in 1984 [3]. The parameter that had to be considered was gini impurity versus information gain or entropy. This parameter gives criteria for which the training data can be split during tree creation. A paper comparing the two of them on a theoretical level concluded that the two only disagree in about 2% of the cases and are empirically the same [4]. Due to this, the Gini index was used because it does not use the logarithm function like entropy which is more computationally expensive.
An additional advantage that was accrued from one-hot encoding the dataset earlier was a solution to the "Absent Level Problem". This is a poignant problem in our case due to the wide range of values for color and breed. The test set could easily contain a value that did not show up in the training set. According to the paper, "one-hot encoding is perhaps the most straightforward engineering technique that could be applied to the absent levels problem [...] any uncertainty over where to subsequently send these absent levels would be eliminated by recoding the levels of each categorical predictor into separate dummy predictors."[2]. This means that the dummy variables would allow the model to handle the cases where the categorical value only appears in the test set. This is especially important for a random forest where nodes are split based on the categorical values.
The optimal hyperparameters were found to be 10,000 estimators, the Gini Index as the node evaluation criteria, the oob_score value set to true, and a minimum of 5 samples per leaf.

3.5.2 Results

After training the Random Forest the accuracy of the model was 0.629. The model did not predict any Euthanasia or deaths. Everything the model predicted is seen in Figure 8.

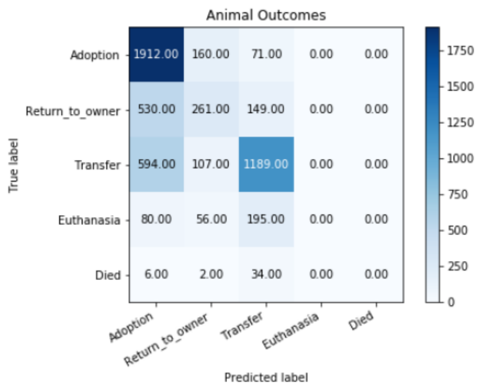


Figure 8: The Confusion matrix of the Random Forest

3.6 Multilayer Perceptron

3.6.1 Description

There were not high hopes for a neural network giving better results than the other methods primarily due to the lack of relational structure in the data. This lack of spatial structure was also the reasoning behind avoiding a convolutional neural network. If the dataset contained images of dogs and cats it might have been a better pick. Two hidden layers with 16 and 12 neurons respectively combined with the relu activation function were found to be optimal.

3.6.2 Results

The Multilayer Perceptron is trained with two different layers with 16 and 12 neurons, and combined with the Relu activation function. After training in this model an accuracy of 0.621 was found.

4 Discussion and Conclusion

The accuracy of the different models lies between 0.541 and 0.638. The model that gives the best accuracy in predicting the Animal outcome, based on the animal input, is the Multinomial Logistic Regression. In table 1 an overview of all the accuracies can be found. There can be made some adjustments in the research to improve the accuracy. In the dataset there is a class imbalance. There are lot more animals that get adopted or get a transfer than animals that died. When the models got tested with the data only one model predicted one time died. This was not a good classification. Also the Euthanasia class was too small for the models to make a good prediction for that class.

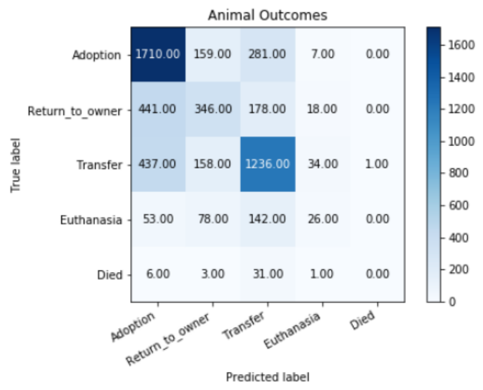


Figure 9: The Confusion matrix of the Multilayer Perceptron

Methods	Accuracy
Naïve bayes	0.541
Multinomial Logistic Regression	0.638
K-nearest neighbors	0.612
Support vector machine	0.633
Random Forest	0.629
Multilayer Perception	0.621

Table 1: Overview of the accuracy of the models with all classes

If both classes were removed, the averages of models got a lot better (see table 2). The experiments were run on both datasets and the accuracy now lies between 0.713 and 0.763. This is a lot higher than the accuracy with both classes. For further research, it would be interesting to look at other possible factors that could influence the outcome of the animals, for instance their name [3].

[3]https://www.datasciencecentral.com/profiles/blogs/predicting-animal-adoption-with-random-forest-svm

Methods	Accuracy
Naïve bayes	0.722
Multinomial Logistic Regression	0.763
Support Vector Machines	0.713
Random Forest	0.757
Multilayer Perception	0.730

Table 2: Overview of the accuracy of the models without the classes Euthanasia and Died

References

[1] Austin Animal Center. URL: http://www.austintexas.gov/department/aac

[2] Timothy C. Au. Random Forests, Decision Trees, and Categorical Predictors: The 'Absent Levels' Problem. Google LLC, 2018. URL: http://jmlr.org/papers/volume19/16-474/16-474.pdf

[3] Leo Breiman. RANDOM FORESTS. University of California Berkeley, 2001. URL: https://www.stat.berkeley.edu/ breiman/randomforest2001.pdf

[4] Laura Elena Raileanu and Kilian Stoffel. Theoretical Comparison between the Gini Index and Information Gain Criteria. University of Neuchâtel, Switzerland, 2004. URL: https://www.unine.ch/files/live/sites/imi/files/shared/documents/papers/Gini_index_fulltext.pdf

[5] Sebastian Raschka, Vahid Mirjalili. Python machine learning : machine learning and deep learning with Python, scikit-learn, and TensorFlow. 2017.

[6] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A Practical Guide to Support Vector Classification. National Taiwan University. 2010. URL: https://www.researchgate.net/profile/Chenghai_Yang/publication/272039161_Evaluating_unsupervised_and_supervised_image_classification_methods_for_mapping_cotton_root_rot/links/55f2c57408ae0960a3897985/Evaluating-unsupervised-and-supervised-image-classification-methods-for-mapping-cotton-root-rot.pdf

[7] Irina Rish. An empirical study of the naive bayes classifier. In IJCAI Workshop on Empirical Methods in Artificial Intelligence, 2001.

[8] Tom M. Mitchell. Machine Learning (pp 154-199). McGraw-Hill. 1997. URL: http://profsite.um.ac.ir/m̃onsefi/machine-learning/pdf/Machine-Learning-Tom-Mitchell.pdf

[9] Scikit-learn (2018). Naive Bayes. URL: https://scikit-learn.org/stable/modules/naive_bayes.html

[10] David W. Hosmer, et al. Applied Logistic Regression. John Wiley and Sons, 2013.

[11] S. B. Imandoust et al. Int. Journal of Engineering Research and Applications Vol. 3, Issue 5, Sep-Oct 2013, pp.605-610. URL: https://www.ijera.com/papers/Vol3_issue5/DI35605610.pdf

[12] Chas Newkey-Burden. The Dog Directory: Hamlyn All Colour Pet Care. 2009.