# Final Project: Animal Rescue Returns Classification

**Naïve Project Team**

*Rachel Kelley (gtID: 903749193) and Justin Schulberg (gtID: 903737135)*

Table of Contents

[Final Project: Animal Rescue Returns Classification 1](#_Toc90133439)

[Breakdown of Roles 2](#_Toc90133440)

[Problem Statement 2](#_Toc90133441)

[Data Source 2](#_Toc90133442)

[Data Preparation/Cleaning 5](#_Toc90133443)

[COLOR 5](#_Toc90133444)

[GENDER 6](#_Toc90133445)

[AGE AT ADOPTION 6](#_Toc90133446)

[BREED MIXES 6](#_Toc90133447)

[MIX 6](#_Toc90133448)

[WEIGHT 6](#_Toc90133449)

[BEHAVIORAL NOTES 6](#_Toc90133450)

[MEDICAL NOTES 7](#_Toc90133451)

[Dealing with NAs 8](#_Toc90133452)

[Dimensionality Reduction 8](#_Toc90133453)

[Balancing the Distribution of Data 10](#_Toc90133454)

[Analysis, Evaluation, and Final Results 11](#_Toc90133455)

# Breakdown of Roles

For the most part, the tasks in the project were equally-shared between the two team members, Rachel and Justin. In particular:

|  |  |
| --- | --- |
| Task | Assignee |
| Requirements Gathering | Justin (Lead) & Rachel (Support) |
| Data Preparation | Justin (Lead) & Rachel (Support) |
| Data Cleaning | Rachel (Lead) & Justin (Support) |
| Classification Models + Analysis | Rachel (Lead) & Justin (Support) |

For a more thorough breakdown of the tasks covered by each team member, feel free to access the [*Issues* page of the GitHub project](https://github.com/jschulberg/Dog-Returns/issues), where various tasks are assigned to the team member who completed them.

# Problem Statement

For this final project, the Naïve Project Team will be analyzing a dataset from [a local Animal Rescue](https://www.luckydoganimalrescue.org/about), a nonprofit dedicated to rescuing homeless, neglected, and abandoned animals from euthanasia in kill shelters and getting them adopted into their forever homes. The nonprofit educates the community and all pet parents on responsible pet parenting, including the importance of spay/neuter, obedience training, and good nutrition.

Successful adoptions of cats and dogs depend on a variety of factors, both on the parts of the pet and of the family. The nonprofit helps bridge the gap, building strong relationships between the two. The nonprofit rescues hundreds of animals every year, provides them with loving temporary care, and finds them well-matched, carefully screened forever homes.

One of the struggles that the nonprofit runs into is pets being returned after adoption. Even though they adopt out ~2000 dogs per year, about 10% get returned for a variety of factors. The nonprofit assiduously tracks information on all their adoptions and returns. **In this project,** **the** **Naïve Project Team will use various analytical methods learned in Computational Data Analysis to help the nonprofit predict whether or not a dog that they adopt out will be returned**.

# Data Source

The project will be overseen by the *Program Manager for Volunteers and Data Integrity* at the nonprofit. She will provide the data needed for this analysis to the Naïve Project Team. Data will be provided across various spreadsheets, detailing the adoptions and returns per year (i.e. Adoptions 2020, Returns 2018, etc.) from the nonprofit, both of which contain records over the past decade, and can be easily linked by a unique ID field for each dog represented in each dataset. The team has worked to programmatically concatenate the various adoption and returns spreadsheets from various years, which unfortunately are not always in the same format and sometimes have different column headers, into two main data sources:

* **Dog List** | A list of every dog that has been adopted out by the nonprofit over the past 10 years. This list includes a variety of features describing each dog, where each row corresponds to a dog being adopted out, and each column represents a different attribute related to that dog. This dataset is composed of 10 spreadsheets, one for every year over the past decade. Each spreadsheet has records of ~2000 adoptions per year. The attributes in this dataset are as follows:

|  |  |
| --- | --- |
| Field | Description |
| Dog Name | Name of the adopted dog |
| ID | Unique ID corresponding to that dog |
| Link | Link to dog’s profile on the nonprofit site |
| Foster/Boarding | Type of adoption for the dog (short vs. long-term) |
| Sex | Gender of dog |
| Age | Estimated age (sometimes a range if actual age not known) of dog at time of adoption |
| Weight | Estimated weight of dog at time of adoption |
| Breed Mixes | Type of dog |
| Color | Color of dog |
| Behavioral Notes | Free text field describing the behavior of the dog. There’s some consistency in entries depending on the individual entering this field, along with some key words to describe the dogs behavior around others |
| Dogs in Home | Number of other dogs in adoptee’s house, if any at all |
| Cats in Home | Number of cats in adoptee’s house, if any at all |
| Kids | Number of kids in adoptee’s house, if any at all |
| BS/W | Indicator for whether or not the dog is part pitbull or other ‘bully’ breed |
| Medical Notes | Free text field describing any health conditions of the dog, if any exist at all |
| Transport Date | Date of adoption |

* **Returns List** | A list of every dog that’s been returned *after being adopted out* by the nonprofit. This list also includes a variety of features describing each dog, where each row corresponds to a dog being adopted out, and each column represents a different attribute related to that dog. This dataset is composed of 10 spreadsheets, one for every year over the past decade. Each spreadsheet has records of ~200 returns per year.

|  |  |
| --- | --- |
| Field | Description |
| Dog Name | Name of the adopted dog |
| ID | Unique ID corresponding to that dog |
| Dog Info | Free text field with general information on the dog |
| Reason for Return | Free text field describing the reason the dog was returned by the adopters. There’s some consistency in this field, but it generally depends on the person entering the information into the spreadsheet |
| Behavior with Dogs | Free text field describing the dog’s general behavior around other dogs |
| Behavior with Kids | Free text field describing the dog’s general behavior around kids |
| Behavior with Cats | Free text field describing the dog’s general behavior around cats |
| Energy Level | General energy of the dog |
| Socialization/Daycare | Categorical field for whether the dog was in a daytime program for dogs |
| Vetting | Categorical field for whether the dog was being taken to the vet |
| Date of Adoption | Date dog was adopted |
| Previous Return? | Boolean for whether the dog has been returned in the past |
| Previous Return Info | Free text field describing why the dog had been returned in the past |
| Date of Return | Date dog was returned |
| Type | Boolean for dog vs. puppy |

# Data Preparation/Cleaning

To begin, we processed, prepared, and cleaned the data for analysis. Currently, all data is kept by the nonprofit in different spreadsheets by year (i.e. adoptions in 2021 are in the ‘Dog List 2021’; so we programmatically combined all of our datasets. On top of that, the Dog and Returns Lists are kept as two separate data sources, but share a linking variable ‘ID’ for dogs that were returned. We used a Left Join to find dogs that were returned; for any dogs which did not have a match in the join, we flagged those as ‘Not Returned’.

There were also some issues with the features provided in the data that needed to be addressed. Some of the work done to properly clean the different features is delineated below:

### [COLOR](https://github.com/jschulberg/Dog-Returns/issues/15)

We created a new column called *COLOR\_FIXED*, which is the cleaned up version of the COLOR column. After cleaning up the COLOR column with *COLOR\_FIXED*, we also made the following new features for predictive purposes:

* **multi\_color** | Variable indicating whether or not a dog has multiple colors denoted in it. This is calculated by seeing if the number of commas (',') in the COLOR\_FIXED column is greater than 1 or whether the color is denoted as 'tri-color'.
* **num\_colors** | Continuous variable counting up the number of commas (',') in the COLOR\_FIXED column.
  + Note: This feature is highly correlated to the feature **multi\_color** since any dog which has num\_colors = 1 🡺 multi\_color = 0; conversely, any dog which has num\_colors > 1 🡺 multi\_color = 1. Though we believe that **num\_colors** will be a better predictor than **multi\_color** because it contains more information, we will test each of these features in our models **separately** to see which one ends up being more predictive.
* **contains\_black** | Variable indicating whether 'black' appears in the COLOR\_FIXED column. This feature, in particular, is being incorporated to test the [hypothesis that black dogs are adopted less/returned more](https://en.wikipedia.org/wiki/Black_dog_syndrome) frequently than other-colored dogs.
* **contains\_white** | Variable indicating whether 'white' appears in the COLOR\_FIXED column.
* **contains\_yellow** | Variable indicating whether 'tan/yellow/golden' appear in the COLOR\_FIXED column.

### [GENDER](https://github.com/jschulberg/Dog-Returns/issues/11)

From the *GENDER* column, due to issues with the manual entry of values, we had to standardize all values as either ‘Female’ or ‘Male’, and then one-hot encoded this; 1 for ‘Male’, 0 for ‘Female.

### [AGE AT ADOPTION](https://github.com/jschulberg/Dog-Returns/issues/10)

In the original data set, age was described in either days, weeks, months, or years, or just a number with no age measure, and some dogs had a listed date of birth. To get the most accurate age for each dog, we calculated age at adoption in days by subtracting the date of birth from the adoption date. Where this information was not available, age at adoption was calculated via the age column and converted to an estimate in days.

### [BREED MIXES](https://github.com/jschulberg/Dog-Returns/issues/13)

We cleaned the *BREED MIXES* column, which denotes a dog's breed. Because there are so many combinations of different breeds for dogs, we opted to create indicator features for the most popular groupings of dogs (Lab/Retriever, Shepherd, or Other).

### [MIX](https://github.com/jschulberg/Dog-Returns/issues/14)

The column, *MIX*, denotes whether a dog is a mixed breed. We cleaned up some of the inconsistencies in the denotations of multiple breeds (i.e. the use of ‘&’, ‘and’, ‘/’, ‘with’, etc.). Also, we leveraged the *SECONDARY BREED* column to fill gaps for nulls. Lastly, we looked at the *BREED MIXES* column for a slash ‘/’ or and ‘&’ to determine if a dog is multiple (mixed) breeds. Lastly, we one-hot encoded this feature; 1 for ‘Mix’, 0 otherwise.

### [WEIGHT](https://github.com/jschulberg/Dog-Returns/issues/12)

The column, *WEIGHT*, denotes the weight, in pounds of a dog. Although this column was more consistent than others, some dogs have their weight as just a number (i.e. 60) while others have their weight along with the unit of measurement (i.e. ‘60lbs’ or ‘60 pounds’).

### [BEHAVIORAL NOTES](https://github.com/jschulberg/Dog-Returns/issues/20)

The column, *BEHAVIORAL NOTES*, includes comments about a dog's temperament. There's not too much cleaning that was performed here -- just some general standardization --, but there were a lot of good notes about dogs' temperaments. To pull out this data, we created the following features:

* **num\_behav\_issues** | Continuous variable denoting the number of behavioral issues/notes a dog has associated with it.
* **puppy\_screen** | Variable indicating whether a dog has a behavioral note that it should be screened with puppies.
* **new\_this\_week** | Variable indicating whether a dog has a behavioral note that it is new in a given week.
  + *Note: We expect this variable to be the least helpful, since a dog being new is not necessarily associated with its adoption date.*
* **needs\_play** | Variable indicating whether a dog has a behavioral note that it doesn’t walk alone enough and needs extra running or playtime.
* **no\_apartments** | Variable indicating whether a dog has a behavioral note that it should not be placed with adopters who live in an apartment, and should instead be placed with adopters who live in a residential home.
* **energetic** | Variable indicating whether a dog has a behavioral note that it exhibits medium-high energy.
* **shyness** | Variable indicating whether a dog has a behavioral note that it exhibits general shyness and should not be placed with kids (who generally expect dogs to be more energetic) and should be committed to a socialization program.
* **needs\_training** | Variable indicating whether a dog has a behavioral note that it should be screened with puppies.

### [MEDICAL NOTES](https://github.com/jschulberg/Dog-Returns/issues/20)

The column, *MEDICAL NOTES*, includes comments about a dog's health conditions. There's not too much cleaning that was performed here -- just some general standardization --, but there were a lot of good notes about the different health conditions that each dog has. We extracted the following features by parsing through the medical notes:

* **has\_med\_issues** | Variable indicating whether a dog has a medical condition associated with it.
* **diarrhea** | Variable indicating whether a dog has a medical note that contains mention of diarrhea.
* **erlichia** | Variable indicating whether a dog has a medical note that contains mention of Ehrlichia (the disease Ehrlichiosis)/Lyme transmitted by ticks.
* **uri** | Variable indicating whether a dog has a medical note that contains mention of Ehrlichia (the disease Ehrlichiosis)/Lyme transmitted by ticks.
* **ear\_infection** | Variable indicating whether a dog has a medical note that contains mention of an ear infection.
* **tapeworm** | Variable indicating whether a dog has a medical note that contains mention of Tapeworm.
* **general\_infection** | Variable indicating whether a dog has a medical note that contains mention of the word ‘infection’.
* **demodex** | Variable indicating whether a dog has a medical note that contains mention of Demodex (Demodectic mange aka mites).
* **car\_sick** | Variable indicating whether a dog has a medical note that contains mention of car sickness.
* **dog\_park** | Variable indicating whether a dog has a medical note that the dog has not yet been exposed to a dog park, indicating a lack of general immunity to common diseases.
* **leg\_issues** | Variable indicating whether a dog has a medical note that contains mention of any leg issues, including amputated, sprained, swollen, lesions, etc.
* **anaplasmosis** | Variable indicating whether a dog has a medical note that contains mention of Anaplasmosis.
* **dental\_issues** | Variable indicating whether a dog has a medical note that contains mention of any teeth/dental issues.
* **weight\_issues** | Variable indicating whether a dog has a medical note that contains mention of weight issues, including overweight and underweight.
* **hair\_loss** | Variable indicating whether a dog has a medical note that contains mention of hair loss.
* **treated\_vaccinated** | Variable indicating whether a dog has a medical note that contains mention of treatment or vaccination for the dog.

## Dealing with NAs

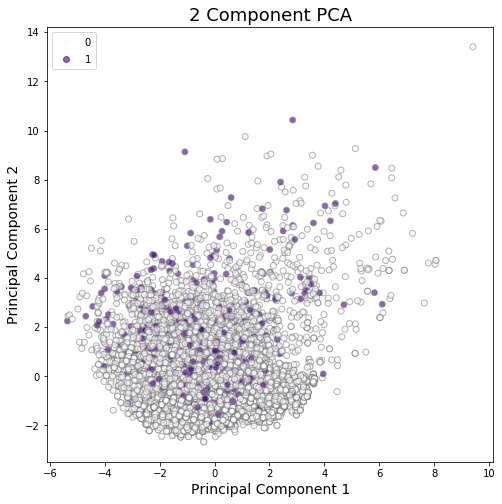
A number of the columns mentioned above have null values. It could be because no medical notes were reported, thus making it impossible to know whether a dog actually has a given medical issue; or because the information was unable to be discerned. To deal with these null values, we had the following options:

* Fill using mean of column
* Fill using median of column
* Fill using clustering algorithm (KNN, K-Means, etc.)

With guidance from the program manager at the nonprofit, we decided to combine a mixture of column fills and KNN imputation to deal with NAs. For column fills, the program manager advised us that some of the features, if not noted, would be 0 – for example, shyness and needs\_play are two features that are generally always noted for dogs with those features, so for the rest of the dogs we can safely assume to fill with zeros. For other columns, such as weight, number of colors, and no\_apartments, we used KNN imputation in order to fill those values, as they are harder to generalize as all 0s or 1s. By using KNN imputation, we are filling the NAs with values appropriate to other features of the dog.

## Dimensionality Reduction

After all the above data cleaning steps were completed, we found ourselves with 43 features! Many of these features were binary variables, while some were continuous variables. To tackle this, we applied PCA on our dataset. Here is what the first two principal components look like, shaded by whether or not a dog was returned:



There’s not much separating out the dogs that were returned (1) from the dogs that stayed adopted (0). Expanding this out to the first three principal dimensions, gives us the following 3-dimensional representation of our principal components:



Again, it’s hard to see much linear structure from these components in regard to the indicator variable. Looking at how much variance in our data is actually explained by each successive principal component, we get:



Looking at this graph, we see that the variation in the data is rather evenly described by each of the principal components. We can see this in the almost linear shape of the graph. That is, for each successive principal component included, about the same amount of variance gets explained. This generally makes sense; most of our features are binary variables. While we are theoretically able to transform the linear space of binary variables, the result is not particularly interpretable since their values are only 0 or 1. The results of this PCA lend this notion credence; going forward, our analysis will be applied directly on the data at hand and not a linear combination of the data.

We also attempted to use [Multiple Correspondence Analysis (MCA)](https://github.com/MaxHalford/prince#multiple-correspondence-analysis-mca), a method similar to PCA that functions better with data that contains *both* numerical and categorical variables. Unfortunately, when running our final classification models later on, we found that applying either PCA or MCA usually resulted in accuracy scores of up to 10% less than the scores when not using these methods.

## Balancing the Distribution of Data

When we started our analysis, 7.8%, or 824 dogs had been ‘returned’ after adoption. That left our majority class of ‘not returned’ dogs at an overweighted distribution of 92.2%, or 9664 dogs. We sought to balance these two classes using resampling techniques before running the classification models.

To do so, we opted to leverage the [Adaptive Synthetic (ADASYN)](https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.ADASYN.html#imblearn.over_sampling.ADASYN) oversampling technique on the ‘returned’ minority class; and a Random undersampling technique on the ‘not returned’ majority class. We found ADASYN, because of its ability to put synthetic points in low distributed areas of the data and then use K-Nearest Neighbors to classify them, to give us the best representation of the minority class. However, we opted not to use ADASYN to get the minority class to the same size (9664) as the majority class; instead opting to increase the size by 3x (2871 ‘returned’ dogs). To get the majority class down in size, we just used a random undersampler to randomly select data points from the majority class (‘not returned’) to keep, ultimately ending with about half as many points (5742 ‘not returned’ dogs).

|  |  |  |  |
| --- | --- | --- | --- |
| Class | # of Original Data Points | | Resampled Data |
| ‘Returned’ | 824 | 2871 | |
| ‘Not Returned’ | 9664 | 5742 | |

Even though this is still not perfectly balanced (i.e. a 1:1 ratio), we contend that this prepares the data in an accurate and unbiased manner for classification. As we tried different techniques and ratios of data points to under/oversample, we applied PCA after the resampling to observe the distribution. Here are the 2-D and 3-D representations of the top three principal components of the resampled data:

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

# Analysis, Evaluation, and Final Results

After preparing the data, we analyzed the data using the various classification models that we have learned so far in class including, but not limited to, Naïve Bayes, K Nearest Neighbors, SVM, Logistic Regression, and Neural Networks. We split the data into training, test, and validation sets. By testing each of these methods, we hope to identify which one is most successful in classifying the adopted dogs as returned vs. not-returned.

## Classification

Here are the final results of the models we tested, with each model’s performance being discussed in further depth below:

|  |  |  |
| --- | --- | --- |
| Model | Performed By | Accuracy |
| Logistic Regression |  |  |
| Naïve Bayes |  |  |
| Random Forest |  |  |
| SVM |  |  |
| Neural Network |  |  |
|  |  |  |
|  |  |  |

To evaluate our models, we looked at their classification/misclassification rates using a confusion matrix, as well as their precision, recall, and F1 scores. These metrics are essential in judging the outcome of a classification model, which is why we will use it to evaluate ours. We will compare the results of our test and validation sets in order to determine which model is the most accurate. We’ll also take note of the performance efficiency of each model.

If our models are accurate, they can be integrated into the nonprofit’s systems to classify new dogs that come into the nonprofit. If a dog is classified into the return group, the nonprofit can ensure that the dog is well matched with its new adopter and reach out to the adopter to provide additional support to prevent the dog from being returned. We plan to construct this project so that the nonprofit’s data team can continue to use it, helping to place more rescued dogs in loving homes and keep them there.

### Naïve Bayes

The Niave Bayes classifier was overall the worst classifier tested. Without any tuning, its accuracy score of 35.2% was mostly brought down by its abysmal specificity score of 2.3%. That is, the classifier predicted that almost all dogs in the test dataset would be returned, even though in actuality only about 10% of dogs will be returned.

After tuning using *GridSearchCV* on the *var\_smoothing* parameter, we were able to increase the accuracy up to 72.2%; however, this still included a lot of key misclassifications in our returned set.

### KNN

For K-Nearest Neighbors, we tested all values of ‘k’ for the odd numbers between 1 and 15. As found in the graph below, we achieved a maximum accuracy when we used a ‘k’ value of 1. That is, the KNN classifier was most accurate when a given test point was only compared to the point nearest to it. This likely has to do with the distribution of our data, which is difficult to parse in a general feature space because of how many of our features are categorical and not continuous, thus making them hard for an algorithm like KNN to properly map and relate different test points together.

Chart, line chart

Description automatically generated

After applying a ‘k’ value of 1 on our entire test dataset, we found a much higher overall accuracy for our dataset:

Chart, treemap chart

Description automatically generated

**Logistic Regression**

Chart, treemap chart

Description automatically generated

**SVM**

Chart, treemap chart

Description automatically generated

**Kernel SVM**

Chart, treemap chart

Description automatically generated

**Neural Networks**

Chart, treemap chart

Description automatically generated

**Decision Tree**

Chart

Description automatically generated

A picture containing diagram

Description automatically generated

**Random Forest**

Chart, treemap chart

Description automatically generated

Chart, histogram

Description automatically generated

**Comparison**

Table

Description automatically generated