Shelter Animals Outcomes

Final Project
W207- Applied Machine Learning
Danielle Adler, Alla Hale, John Pette

For our Kaggle dataset research project, we chose to focus on the Shelter Animal Outcome project at the following link: https://www.kaggle.com/c/shelter-animal-outcomes#description (https://www.kaggle.com/c/shelter-animal-outcomes#description). As our whole team cares about animals and is trying to use data to harness social good, we decided to use this dataset and project to motivate the research question: **"What is the outcome for a shelter animal, based on breed, color, sex, and age?"** While this competition is no longer active, the goal of this exploration would be to identify trends in animal outcomes. Ideally, if we saw patterns, we could then provide recommendations to shelters on animals that seemed to have a higher likelihood of some outcomes over others. The animal features within this dataset are Name, Date of Outcome, Breed, Color, Gender, and Spayed/Neutered vs. Intact. We will use these features to predict which outcome is the most likely for our animals: Return to Owner, Death, Euthanasia, Transfer, or Adoption.

Data Import and Set up

First, we import our libraries.

In [1]:
```
import re
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
import seaborn as sns
import xgboost as xgb

from collections import Counter
import sklearn

# Scikit-learn libraries for model building
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.decomposition import PCA

# Scikit-learn libraries for evaluation
from sklearn.metrics import confusion_matrix
from sklearn.metrics import log_loss
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.model_selection import KFold
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.model_selection import *

# Imbalanced learn libraries to handle unbalanced data
from imblearn.over_sampling import *
from imblearn.under_sampling import *

# Ignoring all warnings throughout the notebook
import warnings
warnings.filterwarnings('ignore')

sklearn.__version__
```

Out[1]:
```
'0.20.1'
```

Next, we load our data, which had already been separated into training and test data. We begin by looking at the structure of the data.

In [2]:
```
# Storing data into training and test dataframes
train_df = pd.read_csv('data/train.csv')
test_df = pd.read_csv('data/test.csv')
train_df.head()
```

Out[2]:

	AnimalID	Name	DateTime	OutcomeType	OutcomeSubtype	AnimalType	SexuponOutco
0	A671945	Hambone	2014-02-12 18:22:00	Return_to_owner	NaN	Dog	Neutered M
1	A656520	Emily	2013-10-13 12:44:00	Euthanasia	Suffering	Cat	Spayed Fem
2	A686464	Pearce	2015-01-31 12:28:00	Adoption	Foster	Dog	Neutered M
3	A683430	NaN	2014-07-11 19:09:00	Transfer	Partner	Cat	Intact M
4	A667013	NaN	2013-11-15 12:52:00	Transfer	Partner	Dog	Neutered M

The competition asked for a very specific submission file. We will load a sample submission file to examine the structure.

In [3]:
```
sample_sub = pd.read_csv('data/sample_submission.csv')
sample_sub.head()
```

Out[3]:

	ID	Adoption	Died	Euthanasia	Return_to_owner	Transfer
0	1	1	0	0	0	0
1	2	1	0	0	0	0
2	3	1	0	0	0	0
3	4	1	0	0	0	0
4	5	1	0	0	0	0

Exploratory Data Analysis (EDA)

To begin our work, we start with exploratory data analysis. First, we look to see if we have outliers or missing data for each feature.

In [4]:
```
# Describe the dataset.
train_df.describe()
```

Out[4]:

	AnimalID	Name	DateTime	OutcomeType	OutcomeSubtype	AnimalType	SexuponOutco
count	26729	19038	26729	26729	13117	26729	267
unique	26729	6374	22918	5	16	2	
top	A683269	Max	2015-08-11 00:00:00	Adoption	Partner	Dog	Neutered M
freq	1	136	19	10769	7816	15595	97

In [5]:
```
train_df_counts = pd.DataFrame({'Column Count' : train_df.count(),
   ....:                        '% Percent of Total Data' : train_df.count().div
(max(train_df.count())) * 100})

train_df_counts.round(1)
```

Out[5]:

	% Percent of Total Data	Column Count
AnimalID	100.0	26729
Name	71.2	19038
DateTime	100.0	26729
OutcomeType	100.0	26729
OutcomeSubtype	49.1	13117
AnimalType	100.0	26729
SexuponOutcome	100.0	26728
AgeuponOutcome	99.9	26711
Breed	100.0	26729
Color	100.0	26729

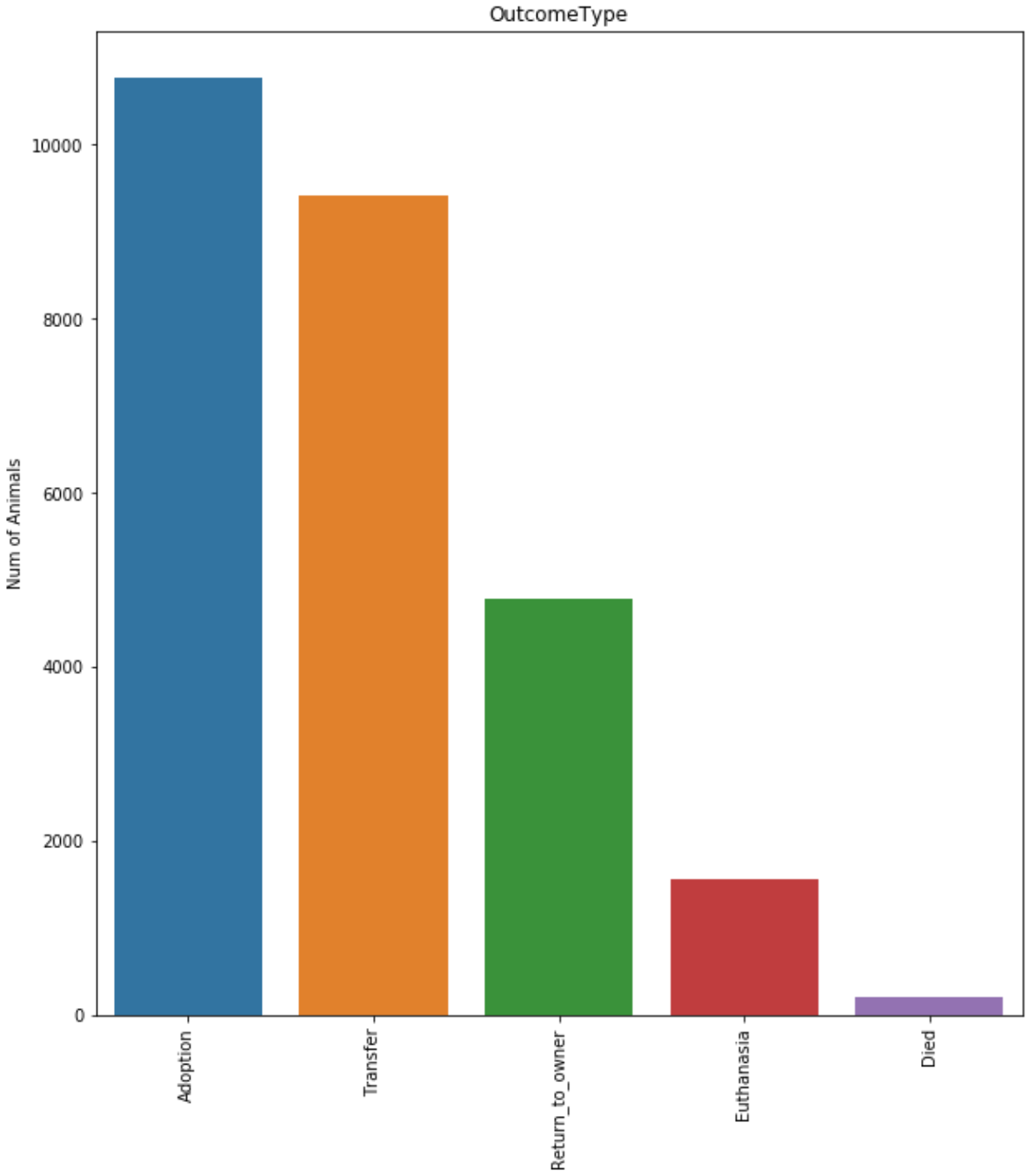
Our initial look at the data reveals 26,729 rows. `Name` is not included for 7,000 cases, but is included for 71.2% of the data. A large quantity of `OutcomeSubtype` data is missing, which we worked to quantify below. Aside from these two columns, we had at least 99.9% of the data in every other column.

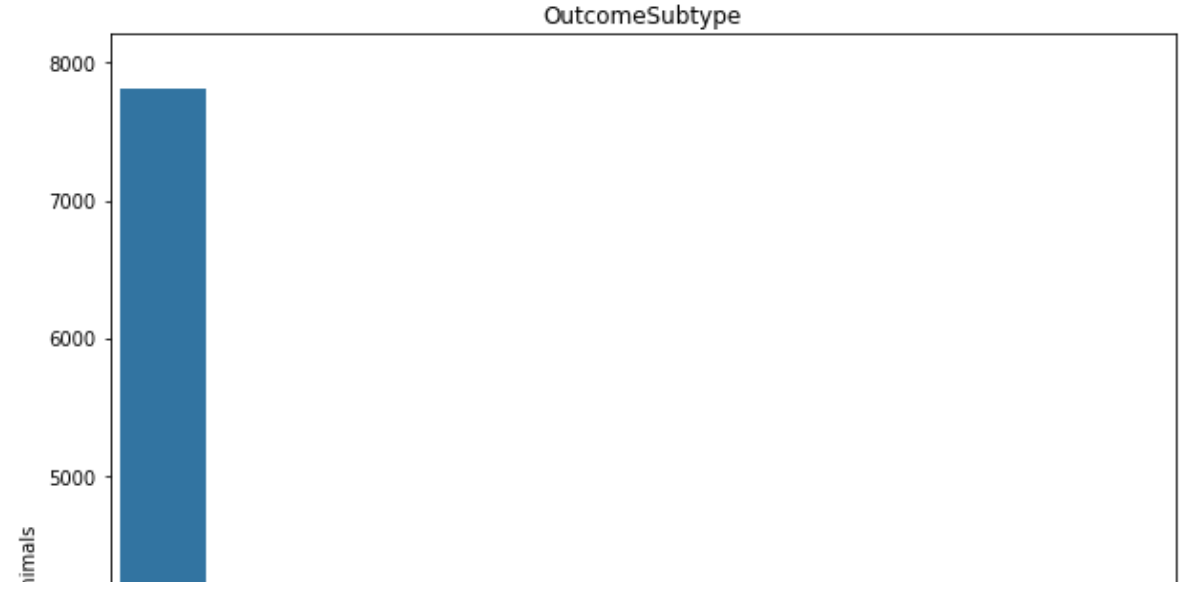
We are choosing not to use the `Name` variable moving forward, as we have blanks for 29% of the data, and animals have a high variety of names, so the variable had few commonalities among animals. We feel that this level of sparsity within `Name` will not help us predict animal outcomes. This makes logical sense as well, as `Name` seems unlikely to be the deciding factor in whether an animal is adopted. Animal owners regularly change animals' names following adoption as well.

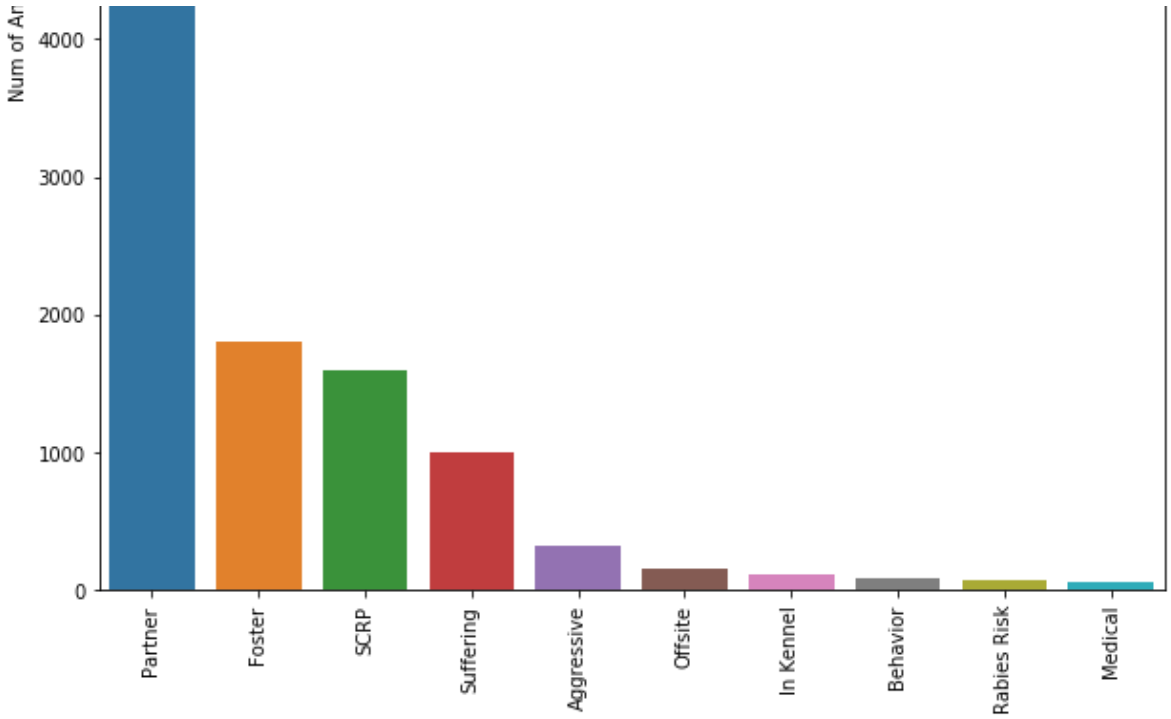
Class Variable Review

In [6]:

```
# Obtain headers.
headers = train_df.dtypes.index

# Categorical headers of classes.
cat_headers = ['OutcomeType', 'OutcomeSubtype']

# Plot paretos of top 10 values in each column.
fig, ax = plt.subplots(len(cat_headers[0:]), 1, figsize=(10, len(cat_headers)*
12))
for i, column in enumerate(cat_headers[0:]):
    to_plot=train_df[column].value_counts().head(10)
    plot=sns.barplot(x=to_plot.index, y=to_plot, ax=ax[i])
    plot.set_xticklabels(to_plot.index, rotation=90)
    plot.set_title(column)
    plot.set_ylabel("Num of Animals")
fig.show()
```


OutcomeType
Num of Animals
10000
8000
6000
4000
2000
0
Adoption
Transfer
Return_to_owner
Euthanasia
Died


OutcomeSubtype
8000
7000
6000
5000



Overall, we see a large imbalance between the outcome classes, which we will discuss further below in our modeling section. The outcomes lean heavily toward Adoption, Transfer, and Return to Owner. There is a smaller, but notable, portion of Euthanasia outcomes, and a very small number of Died outcomes. Partner, which we see below is within the Transfer outcome, is the most common subtype; this presumably means that the animal was transferred to a partner organization or shelter.

While the `Outcome Subtype` variable is also missing for many animals, we do not need to predict based on this variable, so this does not pose a problem for feature evaluation. We cannot use the outcome subtype as a feature, as it would bias the dataset. However, we do want to investigate outcome subtype more fully in an effort to understand the data.

In [7]:
```
train_df.groupby(['OutcomeType', 'OutcomeSubtype']).size().reset_index(name='Count')
```

Out[7]:

	OutcomeType	OutcomeSubtype	Count
0	Adoption	Barn	1
1	Adoption	Foster	1800
2	Adoption	Offsite	165
3	Died	At Vet	4
4	Died	Enroute	8
5	Died	In Foster	52
6	Died	In Kennel	114
7	Died	In Surgery	3
8	Euthanasia	Aggressive	320
9	Euthanasia	Behavior	86
10	Euthanasia	Court/Investigation	6
11	Euthanasia	Medical	66
12	Euthanasia	Rabies Risk	74
13	Euthanasia	Suffering	1002
14	Transfer	Barn	1
15	Transfer	Partner	7816
16	Transfer	SCRP	1599

In exploring `Outcome Subtype`, we note a large quantity of null values in the `Adoption` outcome. The only subtypes labeled in this category are `Foster`, `Offsite`, and in one case, `Barn`. We feel an important next step is to dig further into the null data.

Key `Outcome Subtype` definitions are:

- SCRP = Stray Cat Return Program, a catch-and-release program to spay/neuter stray cats (https://www.maddiesfund.org/austin-animal-services-stray-cat-return-program.htm (https://www.maddiesfund.org/austin-animal-services-stray-cat-return-program.htm))
- Barn = Barn Cat Program, which places cats from euthanasia lists that "cannot be adopted as traditional pets" into safe, appropriate adoption environments, such as a barn, stable, garage, or warehouse. They have access to shelter, food, and water. (https://www.austinpetsalive.org/ (https://www.austinpetsalive.org/) - https://www.austinpetsalive.org/about/programs/barn-cat-placements/ (https://www.austinpetsalive.org/about/programs/barn-cat-placements/))

In [8]:
```
train_df_sub = train_df[train_df['OutcomeSubtype'].isnull()]
train_df_sub.head()
```

Out[8]:

	AnimalID	Name	DateTime	OutcomeType	OutcomeSubtype	AnimalType	SexuponOutc
0	A671945	Hambone	2014-02-12 18:22:00	Return_to_owner	NaN	Dog	Neutered M
8	A671784	Lucy	2014-02-04 17:17:00	Adoption	NaN	Dog	Spayed Fer
11	A666320	NaN	2013-11-04 14:48:00	Adoption	NaN	Dog	Spayed Fer
13	A704702	Scooter	2015-06-08 16:30:00	Return_to_owner	NaN	Dog	Neutered M
14	A688584	Preston	2015-11-25 15:00:00	Return_to_owner	NaN	Dog	Neutered M

From the data in the table above, we surmise that it is possible the majority of the subtype null data is from the `Return to Owner` or `Adoption` outcome type data. We will examine this further below.

In [9]:
```
train_df_sub_2 = train_df_sub['OutcomeType'].value_counts().reset_index(name=
'Count of Null')
train_df_sub_2.rename(index=str, columns={"index": "Outcome Types"})
```

Out[9]:

	Outcome Types	Count of Null
0	Adoption	8803
1	Return_to_owner	4786
2	Died	16
3	Transfer	6
4	Euthanasia	1

8,803 of the Adoption outcome entries have the outcome subtype as `null`. As we already have outcome subtypes for barn, foster, and offsite within adoption, we feel confident in assuming that these missing subtypes are simply the animals that were adopted. We will assume this is true and fill these null values with `adopted`.

As the return-to-owner outcome type also has no subtype, we will make the outcome subtype `return to owner`. It seems that, within the dataset, when the outcome subtype would have been redundant with the outcome type, the outcome subtype was not always listed. Logically, once an animal is adopted or returned to their owner, a shelter feels it does not need further classification or follow up on specific outcomes.

For the remaining 23 null outcome subtypes, we will label their subtype as `other`.

In [10]:
```
def subtype(x):
    if x['OutcomeType'] == 'Adoption' and pd.isnull(x['OutcomeSubtype']) : ret
urn 'Adopted'
    elif x['OutcomeType'] == 'Return_to_owner' and pd.isnull(x['OutcomeSubtyp
e']): return 'Return to Owner'
    elif pd.isnull(x['OutcomeSubtype']) : return 'Other'
    else: return x['OutcomeSubtype']

train_df['OutcomeSubtype'] = train_df.apply(subtype, axis=1)
```

In [11]:
```
train_df.groupby(['OutcomeType', 'OutcomeSubtype']).size().reset_index(name='C
ount')
```

Out[11]:

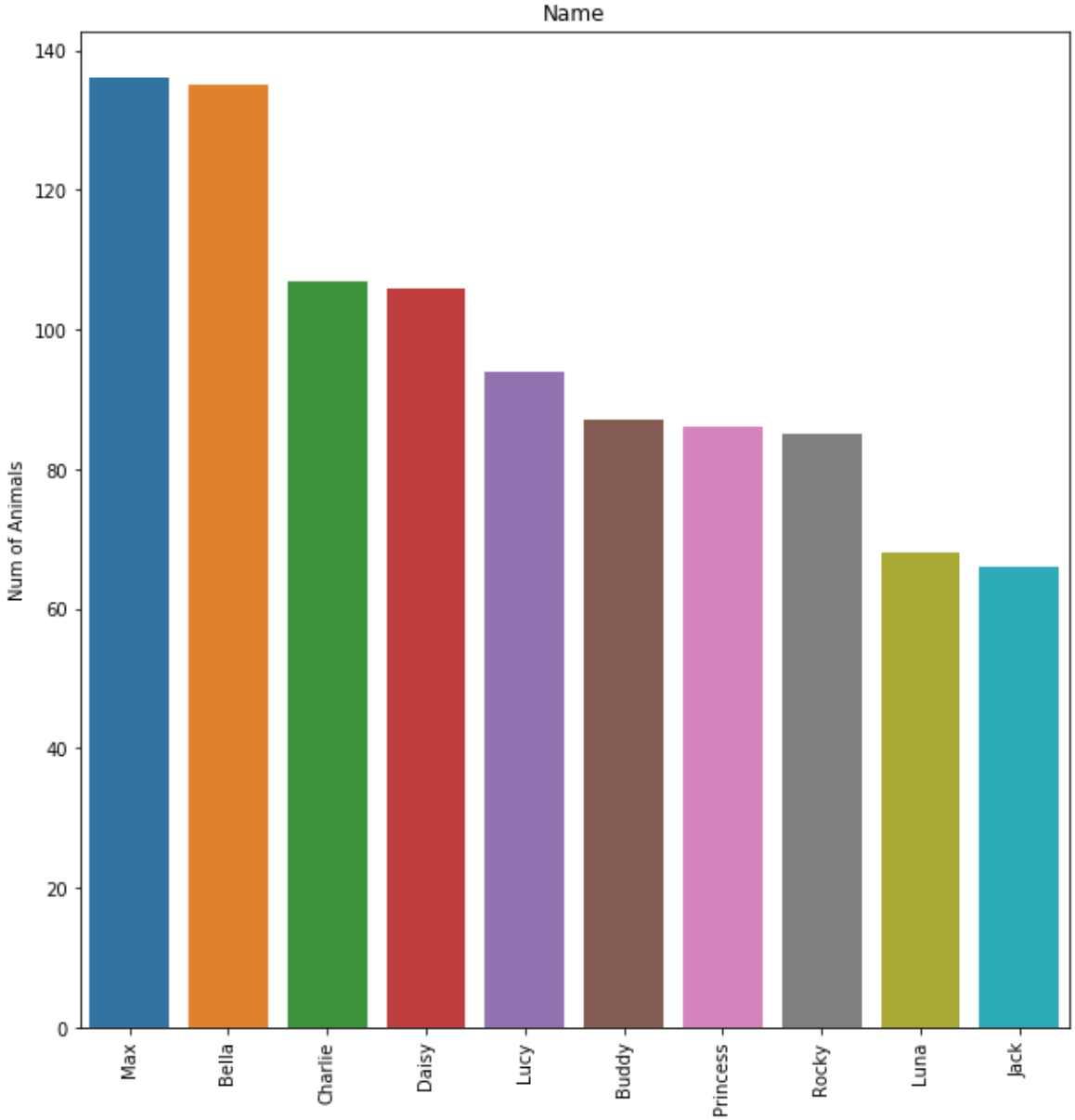
	OutcomeType	OutcomeSubtype	Count
0	Adoption	Adopted	8803
1	Adoption	Barn	1
2	Adoption	Foster	1800
3	Adoption	Offsite	165
4	Died	At Vet	4
5	Died	Enroute	8
6	Died	In Foster	52
7	Died	In Kennel	114
8	Died	In Surgery	3
9	Died	Other	16
10	Euthanasia	Aggressive	320
11	Euthanasia	Behavior	86
12	Euthanasia	Court/Investigation	6
13	Euthanasia	Medical	66
14	Euthanasia	Other	1
15	Euthanasia	Rabies Risk	74
16	Euthanasia	Suffering	1002
17	Return_to_owner	Return to Owner	4786
18	Transfer	Barn	1
19	Transfer	Other	6
20	Transfer	Partner	7816
21	Transfer	SCRP	1599

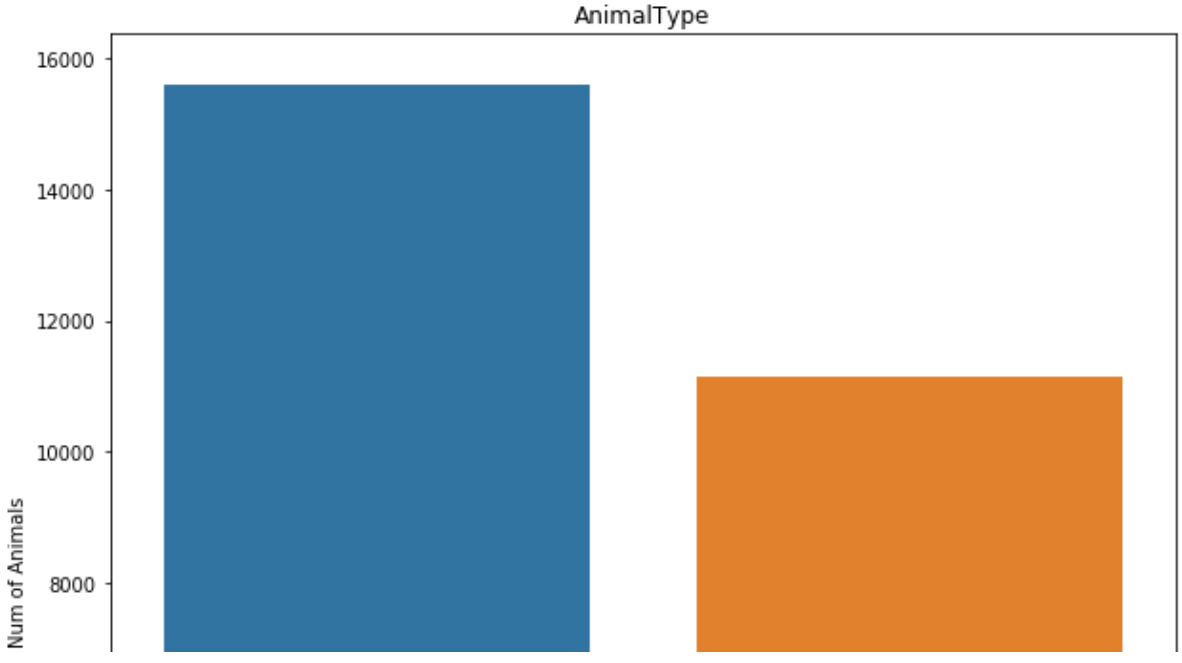
Feature Review

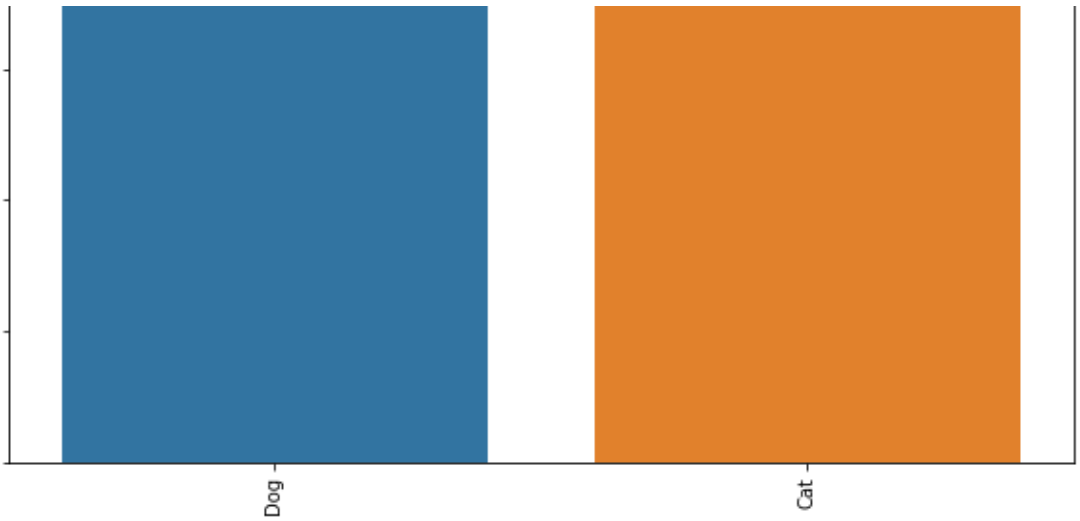
Following the class review, we will now review the specific features themselves. We begin by performing a graphic review of the distribution of values among features in the training set.

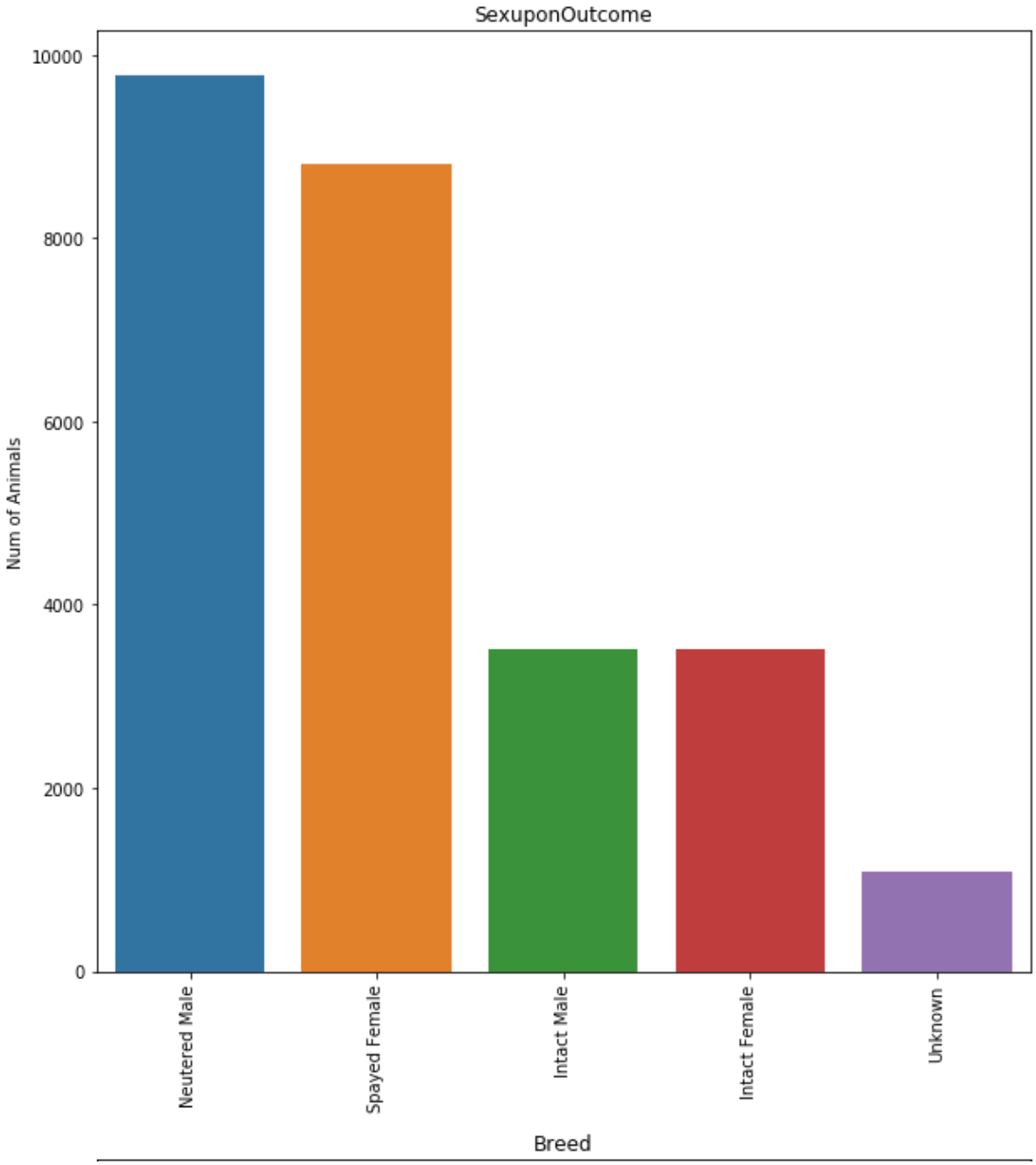
In [12]:

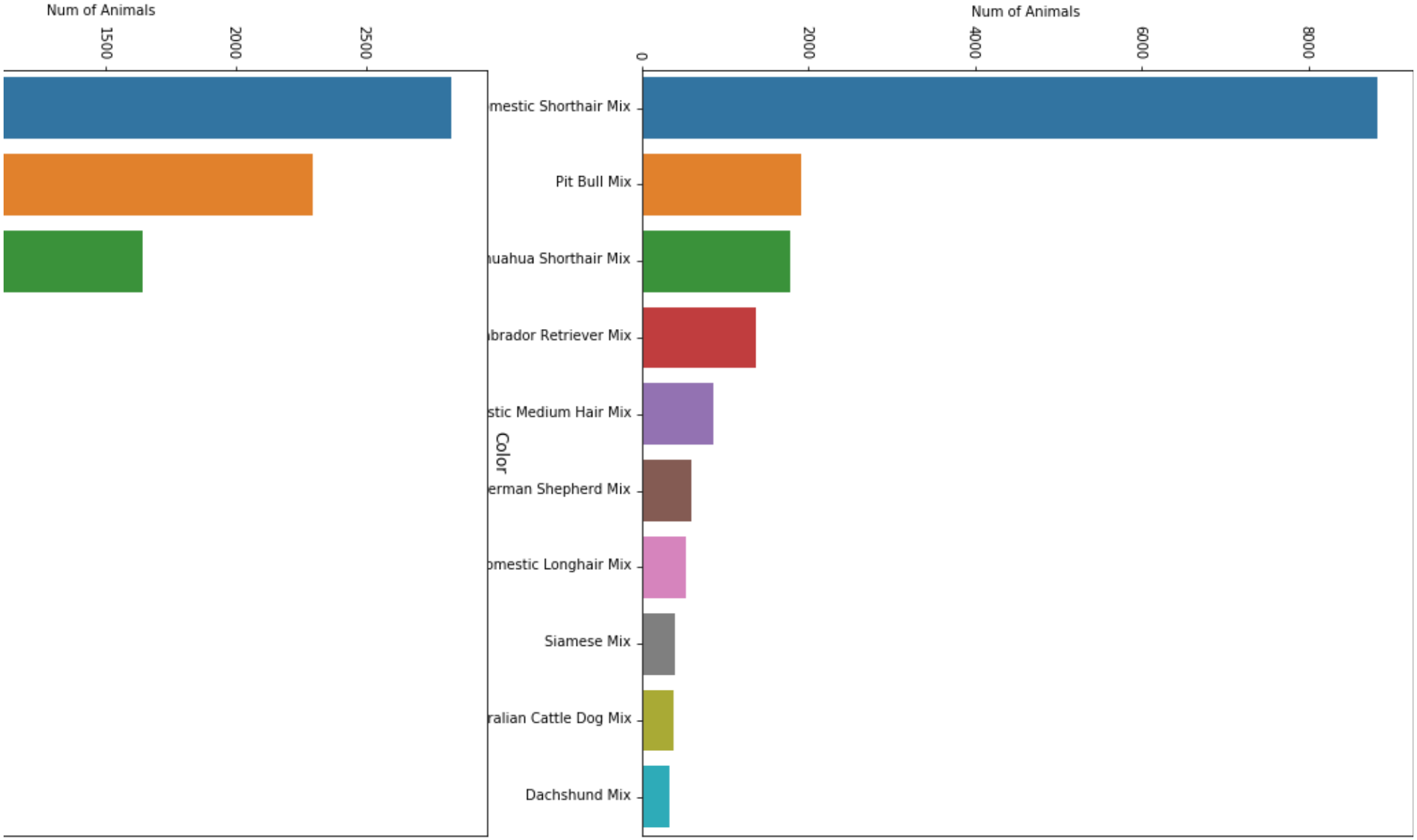
```
# Obtain headers.
headers = train_df.dtypes.index

# Not all of these are categorical variables.
# We need categorical headers and continuous headers, with two different types
of plots.

# Categorical headers of features.
cat_headers = ['Name', 'AnimalType', 'SexuponOutcome', 'Breed', 'Color']

# Plot paretos of top 10 values in each column.
fig, ax = plt.subplots(len(cat_headers[0:]), 1, figsize=(10, len(cat_headers)*
12))
for i, column in enumerate(cat_headers[0:]):
    to_plot=train_df[column].value_counts().head(10)
    plot=sns.barplot(x=to_plot.index, y=to_plot, ax=ax[i])
    plot.set_xticklabels(to_plot.index, rotation=90)
    plot.set_title(column)
    plot.set_ylabel("Num of Animals")
fig.show()
```

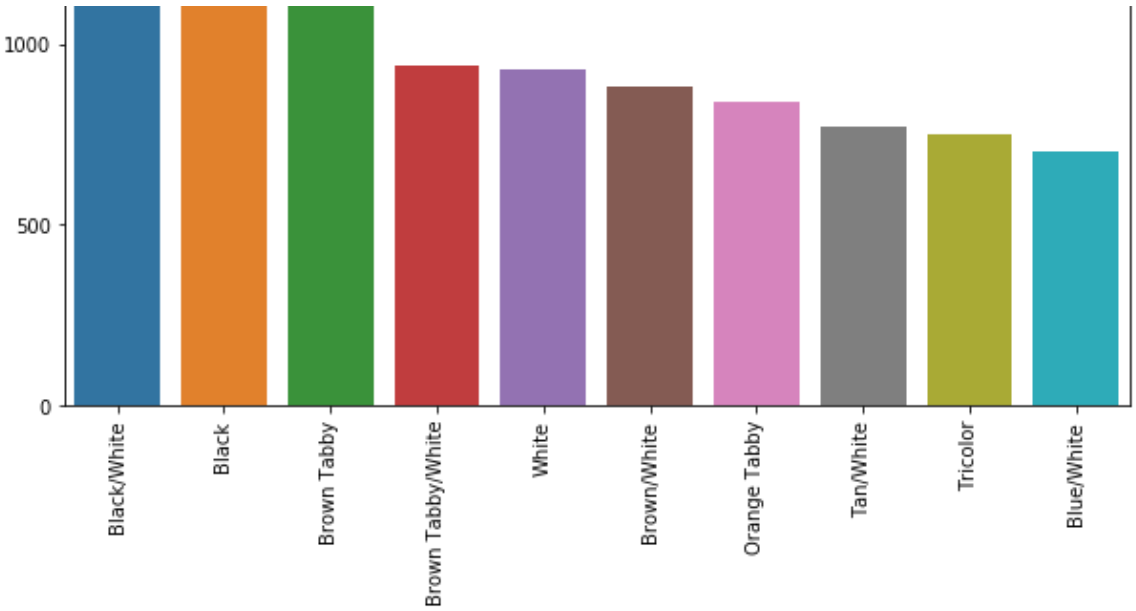

Name
Num of Animals
0
20
40
60
80
100
120
140
Max
Bella
Charlie
Daisy
Lucy
Buddy
Princess
Rocky
Luna
Jack


AnimalType
Num of Animals
8000
10000
12000
14000
16000


6000
4000
2000
0
Dog
Cat


SexuponOutcome
10000
8000
6000
4000
2000
0
Num of Animals
Neutered Male
Spayed Female
Intact Male
Intact Female
Unknown
Breed


Num of Animals
1500
2000
2500
Num of Animals
0
2000
4000
6000
8000
mestic Shorthair Mix
Pit Bull Mix
uahua Shorthair Mix
brador Retriever Mix
stic Medium Hair Mix
Color
erman Shepherd Mix
omestic Longhair Mix
Siamese Mix
ralian Cattle Dog Mix
Dachshund Mix



A breakdown of `Animal Type` shows that there are only dogs and cats in the dataset, with dogs accounting for about 60% of cases.

Sex Upon Outcome shows us whether the animal is male or female and whether they were spayed or neutered (fixed) prior to the outcome. There is a roughly even split between male and female animals, an overwhelming majority of which were spayed or neutered upon outcome. There are also a small, but notable, quantity of unknowns. We will look further below at this variable's interaction with animal outcomes.

The Breed feature is dominated by domestic shorthair cats, which is a catch-all breed label and by far the most common type of cat. There are nearly 9,000 of these. No other breed exceeds 2,000. The most common dog breed is Pitbull Mix.

The most common color is Black/White, followed closely by Black. Taken together as predominantly black, these two colors would substantially outweigh any others. Brown Tabby and Brown Tabby/White are the next most common colors, which is interesting, as these are colors that are only attributable to cats. Black/White and Black could be either dogs or cats. The other top colors are evenly distributed.

In [13]:
```
train_df_sex_dropna = train_df.dropna(subset=['SexuponOutcome'])
train_df_sex_outcome = train_df_sex_dropna.groupby(['OutcomeType', 'SexuponOut
come']).agg({'AnimalType':'size'}).rename(columns={'AnimalType':'# of Animals'
}).reset_index()
train_df_sex_outcome
```

Out[13]:

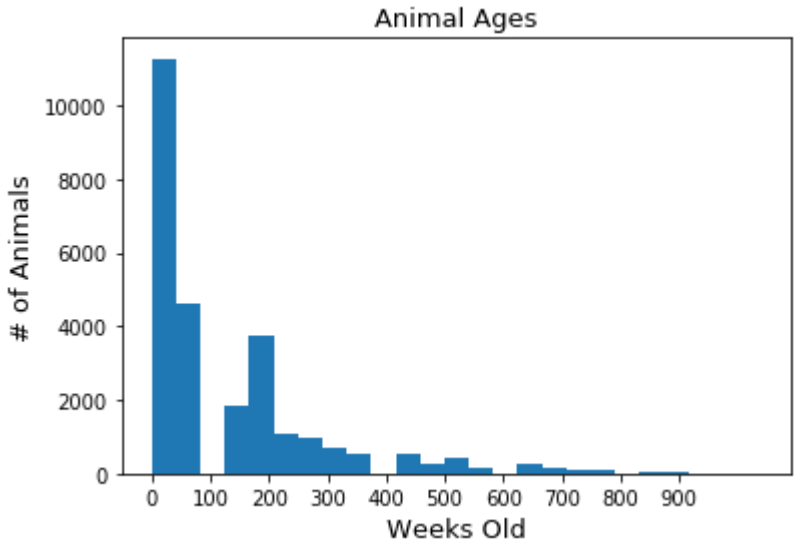
	OutcomeType	SexuponOutcome	# of Animals
0	Adoption	Intact Female	203
1	Adoption	Intact Male	158
2	Adoption	Neutered Male	5222
3	Adoption	Spayed Female	5186
4	Died	Intact Female	56
5	Died	Intact Male	79
6	Died	Neutered Male	19
7	Died	Spayed Female	18
8	Died	Unknown	25
9	Euthanasia	Intact Female	401
10	Euthanasia	Intact Male	477
11	Euthanasia	Neutered Male	344
12	Euthanasia	Spayed Female	232
13	Euthanasia	Unknown	101
14	Return_to_owner	Intact Female	301
15	Return_to_owner	Intact Male	477
16	Return_to_owner	Neutered Male	2247
17	Return_to_owner	Spayed Female	1748
18	Return_to_owner	Unknown	12
19	Transfer	Intact Female	2550
20	Transfer	Intact Male	2334
21	Transfer	Neutered Male	1947
22	Transfer	Spayed Female	1636
23	Transfer	Unknown	955

Examining the distribution of sexes per outcome illuminates the story better than the earlier graphic representation. More than 10,000 of the adopted animals were spayed or neutered upon adoption. Only 361 were not. Of the animals that were returned to owners, a large majority had also been spayed or neutered. Among the other category, spayed/neutered/male/female were all evenly distributed. Most of the unknowns are seen in the transfer outcome. We will look to binarize gender and whether an animal was fixed for our analysis below.

Next, we look into the Age Upon Outcome feature variable. We believed that it is important to make the ages variable continuous, as opposed to categorical, as the relationship of age is continuous in nature. The most logical unit to standardize ages is weeks. Most of the ages are given in weeks, months, or years, all of which easily transform to weeks. Several ages are given in days, but all are 6 days or fewer. We will standardize these as zero weeks so that they are on the same scale as the rest of the data.

In [14]:
```
# Map all AgeuponOutcome values to numbers. Mapped to weeks and rounded to the
nearest week.
train_df['AgeuponOutcome'] = train_df['AgeuponOutcome'].map({'1 week': 1, '1 w
eeks': 1, '2 weeks': 2, '3 weeks': 3,
                                                             '4 weeks': 4, '5
 weeks': 5, '1 month': 4, '2 months': 9,
                                                             '3 months': 13,
'4 months': 17, '5 months': 21,
                                                             '6 months': 26,
'7 months': 30, '8 months': 34,
                                                             '9 months': 38,
'10 months': 43, '11 months': 47,
                                                             '1 year': 52, '2
 years': 204, '3 years': 156,
                                                             '4 years': 208,
'5 years': 260, '6 years': 312,
                                                             '7 years': 364,
'8 years': 416, '9 years': 468,
                                                             '10 years': 520,
'11 years': 572, '12 years': 624,
                                                             '13 years': 676,
'14 years': 728, '15 years': 780,
                                                             '16 years': 832,
'17 years': 884, '18 years': 936,
                                                             '20 years': 1040,
'0 years': 26, '1 day': 0, '2 days': 0,
                                                             '3 days': 0, '4 d
ays': 0, '5 days': 0, '6 days': 0})
```

In [15]:
```
train_df_age_dropna = train_df.dropna(subset=['AgeuponOutcome'])

plt.hist(train_df_age_dropna['AgeuponOutcome'], bins = 25)

plt.title("Animal Ages")
plt.xlabel("Weeks Old")
plt.ylabel("# of Animals")
plt.xticks(np.arange(0, 1000, step=100))

# Font size transformations.
ax = plt.gca()
for item in ([ax.title, ax.xaxis.label, ax.yaxis.label]):
    item.set_fontsize(13)

plt.grid(False)
```



A substantial portion of the animals in the dataset reached their outcomes at under one year of age (52 weeks). We see sizable numbers of outcomes up to the four-year mark (208 weeks). After four years of age, outcomes drop precipitously.

In [16]:
```
train_df_age_dropna_2 = train_df_age_dropna[(train_df_age_dropna.AgeuponOutcom
e <= 104)]
train_df_age_dropna_3 = train_df_age_dropna[(train_df_age_dropna.AgeuponOutcom
e <= 156)]
print("Examining the ages of younger animals further, ", round((train_df_age_d
ropna_2.shape[0] /
                                                                train_df_age_d
ropna.shape[0])*100,2),
      "% of animals are under the age of two years old and ", round((train_df_
age_dropna_3.shape[0] /
                                                              train_df_
age_dropna.shape[0])*100,2),
     "% are under the age of three years old.", sep="")
```

```
Examining the ages of younger animals further, 59.38% of animals are under th
e age of two years old and 66.2% are under the age of three years old.
```

In [17]:
```
train_df_age_outcome = train_df_age_dropna.groupby('OutcomeType').agg({'Animal
Type':'size',
                                                                       'Ageupo
nOutcome':'mean'}).rename(columns={'AnimalType':'# of Animals',

'AgeuponOutcome':'Avg. Age Upon Outcome (in Wk)'}).reset_index()
print("Avg. Age of Animals Upon Outcome:", (round(train_df_age_dropna.AgeuponO
utcome.mean())), "weeks")

train_df_age_outcome.round(1)
```

```
Avg. Age of Animals Upon Outcome: 127 weeks
```

Out[17]:

	OutcomeType	# of Animals	Avg. Age Upon Outcome (in Wk)
0	Adoption	10769	97.7
1	Died	197	79.3
2	Euthanasia	1551	223.2
3	Return_to_owner	4785	233.7
4	Transfer	9406	91.7

From the information above, we can see that a more likely outcome for younger animals is to be transfered or adopted, while older animals are more likely to be returned to their owner or euthanized. While the `Died` outcome has the lowest average age upon outcome, only 0.7% of animals fall into this category. Therefore, we can assume that many of these animals were unhealthy upon birth and died young.

The age variable is explicitly defined as the age upon outcome; we do not know the ages of animals when they entered shelters. We also do not know how long animals were at the shelter prior to their outcome, and we have no insights into animals still in shelters.

Next, we will look at the `DateTime` column to understand a little more about trends over time in the dataset. We will assume that the `DateTime column` references when the outcome of each animal was determined. We will evaluate this information over time by week to group the data.

In [18]:
```
# Convert our DateTime column to the proper format and then group by week.
train_df['DateTime'] = pd.to_datetime(train_df['DateTime']) - pd.to_timedelta(
7, unit='d')
train_df_time = train_df.groupby([pd.Grouper(key='DateTime', freq='W-SUN')]).s
ize().reset_index().sort_values('DateTime')
train_df_time.columns = ['DateTime', 'All']

# Filter for the specific outcome types so that we can later look at the diffe
rences by outcome type.
headers = ['Adoption', 'Transfer', 'Return_to_owner', 'Euthanasia', 'Died']
for Outcome in headers:
    train_df_outcome = train_df[train_df.OutcomeType == Outcome].groupby([pd.G
rouper(key='DateTime', freq='W-SUN')]).size().reset_index().sort_values('DateT
ime')
    train_df_time = pd.merge(train_df_time, train_df_outcome, on='DateTime', h
ow='outer')

# Rename the columns to represent each outcome header.
train_df_time.columns = ['DateTime', 'All', 'Adoption', 'Transfer', 'Return_to
_owner', 'Euthanasia', 'Died']

train_df_time
```

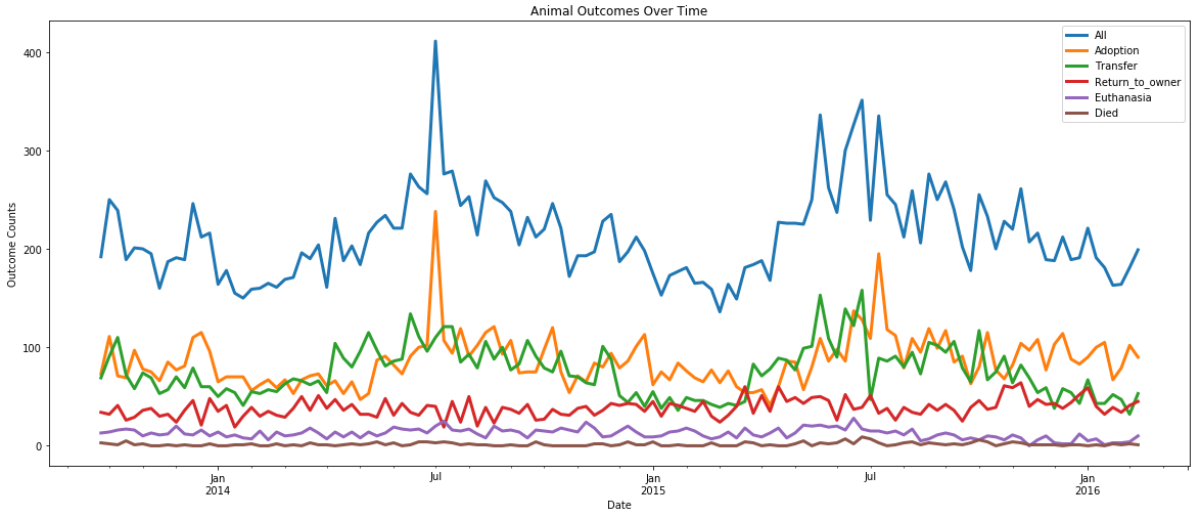
Out[18]:

	DateTime	All	Adoption	Transfer	Return_to_owner	Euthanasia	Died
0	2013-09-29	192	73	69	34	13	3
1	2013-10-06	250	111	91	32	14	2
2	2013-10-13	239	71	110	41	16	1
3	2013-10-20	189	69	72	26	17	5
4	2013-10-27	201	97	58	29	16	1
5	2013-11-03	200	78	74	36	10	2
6	2013-11-10	195	75	69	38	13	0
7	2013-11-17	160	66	53	30	11	0
8	2013-11-24	187	85	57	32	12	1
9	2013-12-01	191	77	70	24	20	0
10	2013-12-08	189	81	59	36	12	1
11	2013-12-15	246	110	79	46	11	0
12	2013-12-22	212	115	60	21	16	0
13	2013-12-29	216	96	60	48	10	2
14	2014-01-05	164	65	50	35	14	0
15	2014-01-12	178	70	58	41	9	0
16	2014-01-19	155	70	54	19	11	1
17	2014-01-26	150	70	41	30	8	1
18	2014-02-02	159	56	55	39	7	2
19	2014-02-09	160	62	53	30	15	0
20	2014-02-16	165	67	57	35	6	0
21	2014-02-23	161	59	55	31	14	2
22	2014-03-02	169	67	63	29	10	0
23	2014-03-09	171	53	68	38	11	1
24	2014-03-16	196	67	66	50	13	0
25	2014-03-23	190	71	62	36	18	3
26	2014-03-30	204	73	66	51	13	1
27	2014-04-06	161	61	54	38	7	1
28	2014-04-13	231	66	104	47	14	0
29	2014-04-20	188	53	89	36	9	1
...	...	...	...	...	...	...	...
95	2015-07-26	245	112	91	26	15	1
96	2015-08-02	212	79	80	39	11	3
97	2015-08-09	259	109	95	34	17	4
98	2015-08-16	206	95	73	32	5	1

	DateTime	All	Adoption	Transfer	Return_to_owner	Euthanasia	Died
99	2015-08-23	276	119	105	42	7	3
100	2015-08-30	250	99	102	36	11	2
101	2015-09-06	268	117	95	42	13	1
102	2015-09-13	240	85	106	36	11	2
103	2015-09-20	202	91	79	25	6	1
104	2015-09-27	178	63	65	39	8	3
105	2015-10-04	255	80	117	46	6	6
106	2015-10-11	233	115	67	37	10	4
107	2015-10-18	200	77	75	39	9	0
108	2015-10-25	228	68	91	61	6	2
109	2015-11-01	220	82	64	59	11	4
110	2015-11-08	261	104	82	64	8	3
111	2015-11-15	207	97	69	40	0	1
112	2015-11-22	216	108	54	47	6	1
113	2015-11-29	189	77	59	42	10	1
114	2015-12-06	188	103	38	43	3	1
115	2015-12-13	212	114	58	38	2	0
116	2015-12-20	189	88	54	44	2	1
117	2015-12-27	191	83	43	52	12	1
118	2016-01-03	221	90	67	59	5	0
119	2016-01-10	191	100	43	40	7	1
120	2016-01-17	181	105	43	32	1	0
121	2016-01-24	163	67	52	39	3	2
122	2016-01-31	164	79	47	34	3	1
123	2016-02-07	181	102	32	41	4	2
124	2016-02-14	199	90	53	45	10	1

125 rows × 7 columns

In [19]:
```
fig, ax = plt.subplots(figsize=(20, 8))
ax = train_df_time.iloc[: , 1:].plot(ax = ax, x=train_df_time['DateTime'], sor
t_columns=True, linewidth=3.0)

plt.xlabel('Date')
plt.ylabel('Outcome Counts')
plt.title('Animal Outcomes Over Time')
ax.legend()
plt.show()
```



From the `DateTime` analysis, we are able to evaluate animal outcomes from Oct 2014 through Feb 2016. We see that most animals are adopted in the summertime, around July and August. Adoption outcomes spike a bit during this time period as well; however, all other outcomes are evenly distributed throughout the time period of analysis.

Feature Manipulation and Binarization

Combining Features

Before binarizing features, we thought doing some simple feature selection could be a good first start. As the breed type has over 1,000 variations and color has over 300 variations, we will see if combining any like features could make sense.

In [20]:

```
# Evaluate the Breed feature set for combination.
train_df_breed_list = train_df['Breed'].tolist()
train_df_breed_dict = {i:train_df_breed_list.count(i) for i in set(train_df_br
eed_list)}

print("Number of unique breeds within the dataset:", train_df['Breed'].describ
e()[1])

# Analyze the different combinations of breeds.
for k in sorted(train_df_breed_dict):
    print(k,": ",train_df_breed_dict[k], sep="", end=", ")
```

Number of unique breeds within the dataset: 1380
Abyssinian Mix: 2, Affenpinscher Mix: 6, Afghan Hound Mix: 1, Airedale Terrie
r: 1, Airedale Terrier Mix: 5, Airedale Terrier/Labrador Retriever: 1, Aireda
le Terrier/Miniature Schnauzer: 1, Akita: 3, Akita Mix: 11, Akita/Australian
Cattle Dog: 1, Akita/Chow Chow: 1, Akita/German Shepherd: 1, Akita/Labrador R
etriever: 1, Akita/Pit Bull: 1, Akita/Siberian Husky: 2, Alaskan Husky: 2, Al
askan Husky Mix: 10, Alaskan Husky/Australian Shepherd: 2, Alaskan Husky/Bord
er Collie: 1, Alaskan Husky/German Shepherd: 1, Alaskan Malamute: 1, Alaskan
Malamute Mix: 5, Alaskan Malamute/Akita: 1, Alaskan Malamute/Australian Kelpi
e: 1, Alaskan Malamute/Border Collie: 2, Alaskan Malamute/German Shepherd: 1,
American Bulldog: 6, American Bulldog Mix: 109, American Bulldog/American Sta
ffordshire Terrier: 1, American Bulldog/Chinese Sharpei: 1, American Bulldog/
Great Pyrenees: 1, American Bulldog/Labrador Retriever: 3, American Bulldog/M
astiff: 2, American Bulldog/Pit Bull: 4, American Bulldog/Pointer: 2, America
n Bulldog/Schnauzer Giant: 1, American Eskimo: 2, American Eskimo Mix: 9, Ame
rican Eskimo/Australian Shepherd: 1, American Eskimo/Border Collie: 2, Americ
an Foxhound: 1, American Foxhound Mix: 2, American Foxhound/Labrador Retrieve
r: 1, American Foxhound/Pointer: 1, American Pit Bull Terrier: 5, American Pi
t Bull Terrier Mix: 68, American Pit Bull Terrier/American Bulldog: 1, Americ
an Pit Bull Terrier/American Pit Bull Terrier: 1, American Pit Bull Terrier/B
oxer: 2, American Pit Bull Terrier/Catahoula: 2, American Pit Bull Terrier/Ch
inese Sharpei: 1, American Pit Bull Terrier/Labrador Retriever: 2, American P
it Bull Terrier/Pointer: 1, American Shorthair Mix: 9, American Staffordshire
Terrier: 12, American Staffordshire Terrier Mix: 92, American Staffordshire T
errier/American Bulldog: 1, American Staffordshire Terrier/French Bulldog: 1,
American Staffordshire Terrier/Labrador Retriever: 3, American Staffordshire
Terrier/Plott Hound: 1, Anatol Shepherd: 12, Anatol Shepherd Mix: 76, Anatol
Shepherd/Australian Cattle Dog: 2, Anatol Shepherd/Australian Shepherd: 1, An
atol Shepherd/Border Collie: 4, Anatol Shepherd/Catahoula: 2, Anatol Shepher
d/Collie Rough: 1, Anatol Shepherd/German Shepherd: 1, Anatol Shepherd/Great
Pyrenees: 6, Anatol Shepherd/Labrador Retriever: 8, Anatol Shepherd/Redbone H
ound: 4, Anatol Shepherd/Siberian Husky: 1, Angora Mix: 7, Australian Cattle
Dog: 25, Australian Cattle Dog Mix: 367, Australian Cattle Dog/Akita: 2, Aust
ralian Cattle Dog/Alaskan Husky: 2, Australian Cattle Dog/American Foxhound:
1, Australian Cattle Dog/American Staffordshire Terrier: 1, Australian Cattle
Dog/Anatol Shepherd: 2, Australian Cattle Dog/Australian Shepherd: 2, Austral
ian Cattle Dog/Basenji: 1, Australian Cattle Dog/Beagle: 5, Australian Cattle
Dog/Belgian Malinois: 1, Australian Cattle Dog/Bloodhound: 2, Australian Catt
le Dog/Border Collie: 10, Australian Cattle Dog/Boxer: 3, Australian Cattle D
og/Bull Terrier: 2, Australian Cattle Dog/Cardigan Welsh Corgi: 3, Australian
Cattle Dog/Catahoula: 3, Australian Cattle Dog/Chinese Sharpei: 2, Australian
Cattle Dog/Chow Chow: 3, Australian Cattle Dog/Collie Rough: 1, Australian Ca
ttle Dog/Collie Smooth: 1, Australian Cattle Dog/Dachshund: 1, Australian Cat
tle Dog/Dalmatian: 1, Australian Cattle Dog/German Shepherd: 8, Australian Ca
ttle Dog/German Shorthair Pointer: 1, Australian Cattle Dog/Golden Retriever:
1, Australian Cattle Dog/Great Pyrenees: 1, Australian Cattle Dog/Jack Russel
l Terrier: 3, Australian Cattle Dog/Labrador Retriever: 31, Australian Cattle
Dog/Parson Russell Terrier: 1, Australian Cattle Dog/Pembroke Welsh Corgi: 2,
Australian Cattle Dog/Pit Bull: 9, Australian Cattle Dog/Plott Hound: 2, Aust
ralian Cattle Dog/Pointer: 5, Australian Cattle Dog/Queensland Heeler: 1, Aus
tralian Cattle Dog/Rat Terrier: 1, Australian Cattle Dog/Rhod Ridgeback: 1, A
ustralian Cattle Dog/Staffordshire: 3, Australian Kelpie: 6, Australian Kelpi
e Mix: 95, Australian Kelpie/Alaskan Husky: 1, Australian Kelpie/American Pit
Bull Terrier: 1, Australian Kelpie/Australian Shepherd: 2, Australian Kelpie/
Blue Lacy: 2, Australian Kelpie/Chihuahua Shorthair: 1, Australian Kelpie/Chi
nese Sharpei: 1, Australian Kelpie/Finnish Spitz: 1, Australian Kelpie/Labrad
or Retriever: 1, Australian Kelpie/Pit Bull: 3, Australian Kelpie/Shiba Inu:

1, Australian Kelpie/Whippet: 1, Australian Shepherd: 14, Australian Shepherd Mix: 163, Australian Shepherd/Alaskan Malamute: 1, Australian Shepherd/Anatol Shepherd: 6, Australian Shepherd/Australian Cattle Dog: 6, Australian Shepherd/Australian Kelpie: 1, Australian Shepherd/Beagle: 1, Australian Shepherd/Border Collie: 6, Australian Shepherd/Boxer: 1, Australian Shepherd/Chihuahua Shorthair: 3, Australian Shepherd/Chow Chow: 3, Australian Shepherd/Cocker Spaniel: 2, Australian Shepherd/Dachshund: 1, Australian Shepherd/English Setter: 1, Australian Shepherd/English Springer Spaniel: 1, Australian Shepherd/Labrador Retriever: 12, Australian Shepherd/Nova Scotia Duck Tolling Retriever: 1, Australian Shepherd/Parson Russell Terrier: 1, Australian Shepherd/Pit Bull: 1, Australian Shepherd/Plott Hound: 1, Australian Shepherd/Rhod Ridgeback: 1, Australian Shepherd/Rottweiler: 1, Australian Shepherd/Siberian Husky: 1, Australian Terrier Mix: 4, Balinese Mix: 5, Basenji Mix: 11, Basenji/Australian Cattle Dog: 2, Basenji/Chihuahua Longhair: 2, Basenji/Chihuahua Shorthair: 1, Basenji/Chow Chow: 1, Basenji/Jack Russell Terrier: 1, Basset Hound: 15, Basset Hound Mix: 43, Basset Hound/Australian Cattle Dog: 2, Basset Hound/Beagle: 2, Basset Hound/Beauceron: 1, Basset Hound/Dachshund: 1, Basset Hound/English Cocker Spaniel: 1, Basset Hound/English Pointer: 1, Basset Hound/Labrador Retriever: 3, Basset Hound/Pit Bull: 1, Basset Hound/Pointer: 1, Beagle: 22, Beagle Mix: 124, Beagle/American Staffordshire Terrier: 1, Beagle/Australian Cattle Dog: 1, Beagle/Australian Kelpie: 1, Beagle/Basset Hound: 4, Beagle/Black Mouth Cur: 1, Beagle/Boston Terrier: 1, Beagle/Boxer: 4, Beagle/Cavalier Span: 1, Beagle/Chihuahua Shorthair: 15, Beagle/Chinese Sharpei: 1, Beagle/Cocker Spaniel: 1, Beagle/Dachshund: 9, Beagle/German Shepherd: 8, Beagle/Jack Russell Terrier: 1, Beagle/Labrador Retriever: 8, Beagle/Manchester Terrier: 1, Beagle/Pembroke Welsh Corgi: 1, Beagle/Pit Bull: 1, Beagle/Pointer: 2, Beagle/Pug: 4, Beagle/Queensland Heeler: 1, Beagle/Rat Terrier: 2, Beagle/Staffordshire: 1, Beagle/Standard Poodle: 2, Beagle/Treeing Walker Coonhound: 1, Beagle/Whippet: 1, Bearded Collie Mix: 2, Beauceron: 1, Beauceron Mix: 11, Bedlington Terr Mix: 3, Belgian Malinois: 2, Belgian Malinois Mix: 15, Belgian Malinois/Australian Shepherd: 1, Belgian Sheepdog: 1, Belgian Sheepdog Mix: 2, Belgian Tervuren Mix: 1, Belgian Tervuren/German Shepherd: 1, Bengal: 2, Bengal Mix: 3, Bernese Mountain Dog: 2, Bernese Mountain Dog Mix: 6, Bernese Mountain Dog/Rottweiler: 1, Bichon Frise: 5, Bichon Frise Mix: 14, Bichon Frise/Miniature Poodle: 1, Black Mouth Cur: 5, Black Mouth Cur Mix: 63, Black Mouth Cur/American Staffordshire Terrier: 1, Black Mouth Cur/Australian Cattle Dog: 3, Black Mouth Cur/Basset Hound: 1, Black Mouth Cur/Blue Lacy: 1, Black Mouth Cur/Boxer: 2, Black Mouth Cur/Catahoula: 4, Black Mouth Cur/Dachshund: 2, Black Mouth Cur/German Shepherd: 3, Black Mouth Cur/Labrador Retriever: 2, Black Mouth Cur/Mastiff: 1, Black Mouth Cur/Staffordshire: 1, Black/Tan Hound: 3, Black/Tan Hound Mix: 18, Black/Tan Hound/Black Mouth Cur: 2, Black/Tan Hound/German Shepherd: 1, Black/Tan Hound/Labrador Retriever: 1, Bloodhound: 1, Bloodhound Mix: 7, Blue Lacy: 4, Blue Lacy Mix: 39, Blue Lacy/American Bulldog: 6, Blue Lacy/Australian Kelpie: 3, Blue Lacy/Beagle: 1, Blue Lacy/Collie Smooth: 1, Blue Lacy/Great Dane: 1, Blue Lacy/Labrador Retriever: 1, Blue Lacy/Pit Bull: 3, Bluetick Hound: 1, Bluetick Hound Mix: 5, Bluetick Hound/Treeing Walker Coonhound: 1, Boerboel: 1, Boerboel Mix: 3, Bombay Mix: 5, Border Collie: 17, Border Collie Mix: 229, Border Collie/Akita: 2, Border Collie/Alaskan Husky: 1, Border Collie/Anatol Shepherd: 1, Border Collie/Australian Cattle Dog: 5, Border Collie/Australian Shepherd: 3, Border Collie/Basenji: 2, Border Collie/Beagle: 1, Border Collie/Boxer: 3, Border Collie/Canaan Dog: 1, Border Collie/Catahoula: 2, Border Collie/Chow Chow: 4, Border Collie/Dachshund: 2, Border Collie/Doberman Pinsch: 1, Border Collie/English Pointer: 1, Border Collie/German Shepherd: 7, Border Collie/Great Dane: 1, Border Collie/Great Pyrenees: 9, Border Collie/Greyhound: 1, Border Collie/Labrador Retriever: 29, Border Collie/Mastiff: 1, Border Collie/Parson Russell Terrier: 2, Border Collie/Pit Bull: 2, Border Collie/Plott Hound: 1, Border Collie/Pointer:

1, Border Collie/Queensland Heeler: 1, Border Collie/Rottweiler: 1, Border Co
llie/Shetland Sheepdog: 1, Border Collie/Standard Schnauzer: 1, Border Colli
e/Whippet: 1, Border Terrier: 1, Border Terrier Mix: 30, Border Terrier/Affen
pinscher: 1, Border Terrier/Bruss Griffon: 1, Border Terrier/Cardigan Welsh C
orgi: 1, Border Terrier/Chihuahua Longhair: 1, Border Terrier/Chihuahua Short
hair: 5, Border Terrier/Dachshund: 1, Border Terrier/Dachshund Wirehair: 1, B
order Terrier/Labrador Retriever: 1, Border Terrier/Miniature Poodle: 1, Bord
er Terrier/Miniature Schnauzer: 1, Border Terrier/Pug: 1, Border Terrier/Rat
Terrier: 1, Border Terrier/Shih Tzu: 1, Border Terrier/Soft Coated Wheaten Te
rrier: 1, Border Terrier/West Highland: 1, Border Terrier/Yorkshire Terrier:
3, Boston Terrier: 13, Boston Terrier Mix: 42, Boston Terrier/Beagle: 1, Bost
on Terrier/Chihuahua Shorthair: 6, Boston Terrier/Dachshund: 1, Boston Terrie
r/Labrador Retriever: 1, Boston Terrier/Miniature Schnauzer: 1, Boston Terrie
r/Pit Bull: 1, Boston Terrier/Pug: 1, Boxer: 30, Boxer Mix: 245, Boxer/Akita:
1, Boxer/American Bulldog: 3, Boxer/American Pit Bull Terrier: 1, Boxer/Ameri
can Staffordshire Terrier: 3, Boxer/Australian Cattle Dog: 4, Boxer/Beagle:
4, Boxer/Black Mouth Cur: 3, Boxer/Boston Terrier: 1, Boxer/Bulldog: 1, Boxe
r/Catahoula: 6, Boxer/Chinese Sharpei: 2, Boxer/Chow Chow: 1, Boxer/English B
ulldog: 2, Boxer/German Shepherd: 1, Boxer/Great Dane: 2, Boxer/Harrier: 1, B
oxer/Jack Russell Terrier: 1, Boxer/Labrador Retriever: 15, Boxer/Mastiff: 1,
Boxer/Miniature Poodle: 2, Boxer/Neapolitan Mastiff: 1, Boxer/Pit Bull: 20, B
oxer/Pointer: 2, Boxer/Redbone Hound: 1, Boxer/Rhod Ridgeback: 2, Boxer/Staff
ordshire: 4, Boykin Span Mix: 3, Boykin Span/Dachshund: 1, British Shorthair:
1, British Shorthair Mix: 3, Brittany: 2, Brittany Mix: 6, Brittany/Chihuahua
Shorthair: 1, Brittany/Labrador Retriever: 2, Bruss Griffon: 1, Bruss Griffon
Mix: 20, Bruss Griffon/Cairn Terrier: 1, Bruss Griffon/Chihuahua Shorthair:
2, Bruss Griffon/Dachshund: 1, Bruss Griffon/Yorkshire Terrier: 1, Bull Terri
er: 7, Bull Terrier Miniature: 1, Bull Terrier Miniature Mix: 2, Bull Terrier
Mix: 27, Bull Terrier/Australian Kelpie: 1, Bull Terrier/Boxer: 1, Bull Terri
er/Jack Russell Terrier: 1, Bull Terrier/Pit Bull: 1, Bulldog: 1, Bulldog Mi
x: 11, Bulldog/Boston Terrier: 1, Bulldog/Pit Bull: 1, Bullmastiff: 2, Bullma
stiff Mix: 8, Bullmastiff/Boxer: 1, Burmese: 1, Cairn Terrier: 10, Cairn Terr
ier Mix: 102, Cairn Terrier/Affenpinscher: 1, Cairn Terrier/Border Collie: 2,
Cairn Terrier/Chihuahua Longhair: 1, Cairn Terrier/Chihuahua Shorthair: 13, C
airn Terrier/Dachshund: 3, Cairn Terrier/Miniature Poodle: 1, Cairn Terrier/M
iniature Schnauzer: 2, Cairn Terrier/Scottish Terrier: 1, Cairn Terrier/Shih
Tzu: 1, Cairn Terrier/Yorkshire Terrier: 5, Canaan Dog Mix: 3, Canaan Dog/Pit
Bull: 1, Canaan Dog/Redbone Hound: 1, Cane Corso: 3, Cane Corso Mix: 4, Cane
Corso/Mastiff: 1, Cardigan Welsh Corgi: 1, Cardigan Welsh Corgi Mix: 54, Card
igan Welsh Corgi/Australian Cattle Dog: 4, Cardigan Welsh Corgi/Australian Ke
lpie: 1, Cardigan Welsh Corgi/Australian Shepherd: 1, Cardigan Welsh Corgi/Be
agle: 1, Cardigan Welsh Corgi/Cairn Terrier: 1, Cardigan Welsh Corgi/Cardigan
Welsh Corgi: 1, Cardigan Welsh Corgi/Chihuahua Longhair: 1, Cardigan Welsh Co
rgi/Chihuahua Shorthair: 10, Cardigan Welsh Corgi/Cocker Spaniel: 2, Cardigan
Welsh Corgi/English Setter: 1, Cardigan Welsh Corgi/German Shepherd: 1, Cardi
gan Welsh Corgi/Jack Russell Terrier: 1, Cardigan Welsh Corgi/Miniature Pinsc
her: 1, Cardigan Welsh Corgi/Pit Bull: 1, Cardigan Welsh Corgi/Pointer: 1, Ca
rdigan Welsh Corgi/Rat Terrier: 1, Carolina Dog: 1, Carolina Dog Mix: 39, Car
olina Dog/Australian Cattle Dog: 1, Carolina Dog/Chihuahua Shorthair: 4, Caro
lina Dog/Labrador Retriever: 2, Carolina Dog/Pit Bull: 1, Catahoula: 6, Catah
oula Mix: 157, Catahoula/Alaskan Husky: 2, Catahoula/American Bulldog: 1, Cat
ahoula/American Pit Bull Terrier: 1, Catahoula/Anatol Shepherd: 2, Catahoula/
Australian Cattle Dog: 6, Catahoula/Australian Shepherd: 1, Catahoula/Beagle:
2, Catahoula/Black Mouth Cur: 1, Catahoula/Border Collie: 1, Catahoula/Boxer:
2, Catahoula/Bulldog: 1, Catahoula/Cardigan Welsh Corgi: 2, Catahoula/Chinese
Sharpei: 2, Catahoula/Chow Chow: 1, Catahoula/German Shepherd: 1, Catahoula/G
reat Dane: 1, Catahoula/Labrador Retriever: 12, Catahoula/Pit Bull: 5, Cataho

ula/Plott Hound: 3, Catahoula/Pointer: 5, Catahoula/Rottweiler: 1, Catahoula/Staffordshire: 1, Catahoula/Whippet: 1, Cavalier Span: 2, Cavalier Span Mix: 8, Cavalier Span/Jack Russell Terrier: 1, Cavalier Span/Papillon: 1, Chesa Bay Retr: 1, Chesa Bay Retr Mix: 12, Chihuahua Longhair: 9, Chihuahua Longhair Mix: 142, Chihuahua Longhair/Cairn Terrier: 3, Chihuahua Longhair/Cardigan Welsh Corgi: 2, Chihuahua Longhair/Dachshund: 2, Chihuahua Longhair/Italian Greyhound: 1, Chihuahua Longhair/Maltese: 2, Chihuahua Longhair/Papillon: 1, Chihuahua Longhair/Pekingese: 2, Chihuahua Longhair/Pomeranian: 2, Chihuahua Longhair/Pug: 1, Chihuahua Longhair/West Highland: 1, Chihuahua Shorthair: 85, Chihuahua Shorthair Mix: 1766, Chihuahua Shorthair/Affenpinscher: 1, Chihuahua Shorthair/Australian Cattle Dog: 2, Chihuahua Shorthair/Basset Hound: 2, Chihuahua Shorthair/Beagle: 10, Chihuahua Shorthair/Border Terrier: 6, Chihuahua Shorthair/Boston Terrier: 7, Chihuahua Shorthair/Boxer: 1, Chihuahua Shorthair/Cairn Terrier: 10, Chihuahua Shorthair/Cardigan Welsh Corgi: 18, Chihuahua Shorthair/Cavalier Span: 1, Chihuahua Shorthair/Chinese Sharpei: 2, Chihuahua Shorthair/Cocker Spaniel: 1, Chihuahua Shorthair/Dachshund: 98, Chihuahua Shorthair/Dachshund Wirehair: 4, Chihuahua Shorthair/Finnish Spitz: 1, Chihuahua Shorthair/French Bulldog: 1, Chihuahua Shorthair/German Shepherd: 2, Chihuahua Shorthair/Italian Greyhound: 6, Chihuahua Shorthair/Jack Russell Terrier: 24, Chihuahua Shorthair/Lhasa Apso: 1, Chihuahua Shorthair/Manchester Terrier: 3, Chihuahua Shorthair/Miniature Pinscher: 9, Chihuahua Shorthair/Miniature Schnauzer: 5, Chihuahua Shorthair/Parson Russell Terrier: 1, Chihuahua Shorthair/Pekingese: 1, Chihuahua Shorthair/Pembroke Welsh Corgi: 1, Chihuahua Shorthair/Pit Bull: 4, Chihuahua Shorthair/Pomeranian: 7, Chihuahua Shorthair/Pug: 18, Chihuahua Shorthair/Queensland Heeler: 1, Chihuahua Shorthair/Rat Terrier: 22, Chihuahua Shorthair/Schipperke: 1, Chihuahua Shorthair/Shetland Sheepdog: 1, Chihuahua Shorthair/Shih Tzu: 1, Chihuahua Shorthair/Smooth Fox Terrier: 1, Chihuahua Shorthair/Toy Fox Terrier: 2, Chihuahua Shorthair/Welsh Terrier: 2, Chihuahua Shorthair/Whippet: 2, Chihuahua Shorthair/Wire Hair Fox Terrier: 3, Chihuahua Shorthair/Yorkshire Terrier: 11, Chinese Crested Mix: 3, Chinese Crested/Chihuahua Longhair: 1, Chinese Crested/Papillon: 1, Chinese Sharpei: 8, Chinese Sharpei Mix: 25, Chinese Sharpei/Airedale Terrier: 1, Chinese Sharpei/Australian Cattle Dog: 1, Chinese Sharpei/Basset Hound: 1, Chinese Sharpei/Border Collie: 2, Chinese Sharpei/Great Dane: 1, Chinese Sharpei/Labrador Retriever: 3, Chinese Sharpei/Pit Bull: 1, Chow Chow: 5, Chow Chow Mix: 57, Chow Chow/Australian Cattle Dog: 5, Chow Chow/Australian Kelpie: 1, Chow Chow/Basset Hound: 3, Chow Chow/Border Collie: 1, Chow Chow/Cardigan Welsh Corgi: 1, Chow Chow/Chinese Sharpei: 2, Chow Chow/English Cocker Spaniel: 1, Chow Chow/German Shepherd: 4, Chow Chow/Golden Retriever: 2, Chow Chow/Labrador Retriever: 7, Chow Chow/Pit Bull: 5, Chow Chow/Siberian Husky: 1, Cocker Spaniel: 20, Cocker Spaniel Mix: 44, Cocker Spaniel/Australian Shepherd: 1, Cocker Spaniel/Chihuahua Longhair: 1, Cocker Spaniel/Chihuahua Shorthair: 2, Cocker Spaniel/Dachshund: 2, Cocker Spaniel/Golden Retriever: 1, Cocker Spaniel/Labrador Retriever: 7, Cocker Spaniel/Miniature Poodle: 4, Cocker Spaniel/Standard Poodle: 1, Cocker Spaniel/Toy Poodle: 1, Collie Rough: 1, Collie Rough Mix: 9, Collie Rough/Australian Cattle Dog: 1, Collie Rough/Australian Shepherd: 2, Collie Rough/Chinese Sharpei: 1, Collie Rough/German Shepherd: 1, Collie Rough/Labrador Retriever: 2, Collie Rough/Pembroke Welsh Corgi: 1, Collie Rough/Pointer: 1, Collie Smooth: 1, Collie Smooth Mix: 29, Collie Smooth/Chow Chow: 1, Collie Smooth/German Shepherd: 2, Collie Smooth/Golden Retriever: 1, Collie Smooth/Jack Russell Terrier: 1, Collie Smooth/Labrador Retriever: 7, Collie Smooth/Siberian Husky: 1, Cornish Rex Mix: 1, Cymric Mix: 2, Dachshund: 46, Dachshund Longhair: 9, Dachshund Longhair Mix: 32, Dachshund Longhair/Australian Shepherd: 1, Dachshund Longhair/Cairn Terrier: 1, Dachshund Longhair/Cardigan Welsh Corgi: 2, Dachshund Longhair/Chihuahua Longhair: 1, Dachshund Longhair/Chihuahua Shorthair: 2, Dachshund Longhair/Cocker Spaniel: 2, Dachshund Longhair/Miniature Schnauzer: 2, Dachshund Longhair/Pembroke Welsh Corgi:

1, Dachshund Longhair/Pomeranian: 1, Dachshund Mix: 318, Dachshund Wirehair: 3, Dachshund Wirehair Mix: 23, Dachshund Wirehair/Border Terrier: 3, Dachshund Wirehair/Cairn Terrier: 1, Dachshund Wirehair/Chihuahua Longhair: 2, Dachshund Wirehair/Chihuahua Shorthair: 10, Dachshund Wirehair/Miniature Poodle: 7, Dachshund Wirehair/Miniature Schnauzer: 1, Dachshund Wirehair/Norfolk Terrier: 1, Dachshund Wirehair/Pbgv: 1, Dachshund Wirehair/Rat Terrier: 1, Dachshund Wirehair/Standard Poodle: 1, Dachshund Wirehair/Toy Poodle: 2, Dachshund/American Staffordshire Terrier: 1, Dachshund/Australian Cattle Dog: 3, Dachshund/Australian Kelpie: 1, Dachshund/Basset Hound: 1, Dachshund/Beagle: 14, Dachshund/Belgian Malinois: 1, Dachshund/Bloodhound: 2, Dachshund/Bull Terrier: 2, Dachshund/Cardigan Welsh Corgi: 2, Dachshund/Catahoula: 2, Dachshund/Cavalier Span: 1, Dachshund/Chihuahua Longhair: 1, Dachshund/Chihuahua Shorthair: 69, Dachshund/Chinese Sharpei: 1, Dachshund/Dachshund: 1, Dachshund/English Bulldog: 1, Dachshund/English Foxhound: 1, Dachshund/German Pinscher: 1, Dachshund/German Shepherd: 1, Dachshund/Greyhound: 1, Dachshund/Harrier: 2, Dachshund/Havanese: 1, Dachshund/Jack Russell Terrier: 2, Dachshund/Labrador Retriever: 4, Dachshund/Manchester Terrier: 8, Dachshund/Miniature Pinscher: 2, Dachshund/Miniature Poodle: 1, Dachshund/Parson Russell Terrier: 1, Dachshund/Pembroke Welsh Corgi: 1, Dachshund/Pit Bull: 3, Dachshund/Pug: 1, Dachshund/Rat Terrier: 8, Dachshund/Shetland Sheepdog: 1, Dachshund/Shih Tzu: 1, Dachshund/Whippet: 2, Dachshund/Yorkshire Terrier: 1, Dalmatian: 4, Dalmatian Mix: 10, Dalmatian/Australian Cattle Dog: 1, Dalmatian/Boxer: 1, Dalmatian/Labrador Retriever: 1, Devon Rex: 1, Devon Rex Mix: 1, Doberman Pinsch: 16, Doberman Pinsch Mix: 45, Doberman Pinsch/Australian Cattle Dog: 1, Doberman Pinsch/Border Collie: 1, Doberman Pinsch/German Shepherd: 1, Doberman Pinsch/Labrador Retriever: 3, Doberman Pinsch/Pit Bull: 2, Doberman Pinsch/Rottweiler: 1, Doberman Pinsch/Vizsla: 1, Dogo Argentino: 2, Dogo Argentino Mix: 13, Dogo Argentino/Chinese Sharpei: 1, Dogo Argentino/Labrador Retriever: 1, Dogue De Bordeaux Mix: 4, Dogue De Bordeaux/American Bulldog: 1, Domestic Longhair: 23, Domestic Longhair Mix: 520, Domestic Longhair/Persian: 2, Domestic Longhair/Rex: 1, Domestic Longhair/Russian Blue: 1, Domestic Medium Hair: 42, Domestic Medium Hair Mix: 839, Domestic Medium Hair/Siamese: 2, Domestic Shorthair: 143, Domestic Shorthair Mix: 8810, Domestic Shorthair/British Shorthair: 1, Domestic Shorthair/Domestic Medium Hair: 1, Domestic Shorthair/Manx: 1, Domestic Shorthair/Siamese: 2, Dutch Shepherd: 1, Dutch Shepherd Mix: 7, Dutch Shepherd/Anatol Shepherd: 1, English Bulldog: 12, English Bulldog Mix: 24, English Bulldog/American Bulldog: 1, English Bulldog/American Staffordshire Terrier: 1, English Bulldog/Beagle: 1, English Bulldog/Boxer: 1, English Bulldog/Cairn Terrier: 1, English Bulldog/Dachshund: 1, English Bulldog/Pit Bull: 1, English Cocker Spaniel Mix: 1, English Coonhound Mix: 5, English Coonhound/Australian Cattle Dog: 1, English Coonhound/Border Collie: 1, English Coonhound/Italian Greyhound: 1, English Foxhound: 2, English Foxhound Mix: 1, English Pointer: 1, English Pointer Mix: 14, English Setter Mix: 2, English Shepherd: 1, English Shepherd Mix: 2, English Springer Spaniel: 2, English Springer Spaniel Mix: 4, English Springer Spaniel/Beagle: 1, English Springer Spaniel/Cardigan Welsh Corgi: 1, Entlebucher Mix: 1, Exotic Shorthair Mix: 2, Feist Mix: 6, Field Spaniel Mix: 2, Finnish Spitz: 2, Finnish Spitz Mix: 7, Finnish Spitz/Chow Chow: 1, Finnish Spitz/Dachshund: 1, Flat Coat Retriever Mix: 37, Flat Coat Retriever/Australian Cattle Dog: 1, Flat Coat Retriever/German Shepherd: 1, Flat Coat Retriever/Golden Retriever: 1, Flat Coat Retriever/Labrador Retriever: 1, Flat Coat Retriever/Papillon: 5, Flat Coat Retriever/Pit Bull: 2, French Bulldog: 1, French Bulldog Mix: 6, French Bulldog/Chihuahua Shorthair: 2, French Bulldog/English Bulldog: 1, French Bulldog/Miniature Schnauzer: 1, German Pinscher Mix: 1, German Pinscher/Whippet: 1, German Shepherd: 77, German Shepherd Mix: 575, German Shepherd/Akita: 1, German Shepherd/Alaskan Husky: 3, German Shepherd/Alaskan Malamute: 5, German Shepherd/Anatol Shepherd: 2, German Shepherd/Australian Cattle Dog: 9, German Shepherd/Australian Kelpie: 1, German S

hepherd/Australian Shepherd: 2, German Shepherd/Black Mouth Cur: 1, German Sh
epherd/Black/Tan Hound: 1, German Shepherd/Border Collie: 7, German Shepherd/
Boxer: 7, German Shepherd/Bull Terrier: 1, German Shepherd/Cardigan Welsh Cor
gi: 4, German Shepherd/Catahoula: 4, German Shepherd/Chinese Sharpei: 4, Germ
an Shepherd/Chow Chow: 10, German Shepherd/Collie Rough: 4, German Shepherd/C
ollie Smooth: 2, German Shepherd/Doberman Pinsch: 3, German Shepherd/Golden R
etriever: 3, German Shepherd/Great Dane: 1, German Shepherd/Great Pyrenees:
2, German Shepherd/Greyhound: 1, German Shepherd/Labrador Retriever: 64, Germ
an Shepherd/Mastiff: 6, German Shepherd/Nova Scotia Duck Tolling Retriever:
1, German Shepherd/Pembroke Welsh Corgi: 2, German Shepherd/Pit Bull: 3, Germ
an Shepherd/Redbone Hound: 1, German Shepherd/Rottweiler: 7, German Shepherd/
Siberian Husky: 11, German Shepherd/Swedish Vallhund: 1, German Shorthair Poi
nter: 4, German Shorthair Pointer Mix: 19, German Shorthair Pointer/Australia
n Cattle Dog: 1, German Wirehaired Pointer: 2, German Wirehaired Pointer Mix:
3, Glen Of Imaal Mix: 3, Glen Of Imaal/Ibizan Hound: 1, Golden Retriever: 13,
Golden Retriever Mix: 66, Golden Retriever/Akita: 1, Golden Retriever/Austral
ian Cattle Dog: 1, Golden Retriever/Australian Shepherd: 1, Golden Retriever/
Beagle: 1, Golden Retriever/Border Collie: 2, Golden Retriever/Brittany: 1, G
olden Retriever/Chow Chow: 5, Golden Retriever/Dachshund: 1, Golden Retrieve
r/Finnish Spitz: 1, Golden Retriever/German Shepherd: 2, Golden Retriever/Gre
at Pyrenees: 1, Golden Retriever/Labrador Retriever: 5, Golden Retriever/Pemb
roke Welsh Corgi: 1, Golden Retriever/Pit Bull: 1, Golden Retriever/Standard
Poodle: 3, Golden Retriever/Whippet: 1, Great Dane: 9, Great Dane Mix: 21, Gr
eat Dane/Australian Cattle Dog: 1, Great Dane/Boxer: 1, Great Dane/German She
pherd: 1, Great Dane/Labrador Retriever: 6, Great Dane/Mastiff: 2, Great Dan
e/Pit Bull: 1, Great Dane/Staffordshire: 1, Great Pyrenees: 14, Great Pyrenee
s Mix: 86, Great Pyrenees/Anatol Shepherd: 4, Great Pyrenees/Australian Cattl
e Dog: 2, Great Pyrenees/Australian Shepherd: 1, Great Pyrenees/Border Colli
e: 6, Great Pyrenees/Boxer: 1, Great Pyrenees/Catahoula: 2, Great Pyrenees/Co
llie Rough: 1, Great Pyrenees/English Setter: 1, Great Pyrenees/German Shephe
rd: 1, Great Pyrenees/Labrador Retriever: 7, Great Pyrenees/Pit Bull: 1, Grea
t Pyrenees/Pointer: 1, Great Pyrenees/Rottweiler: 1, Great Pyrenees/Siberian
Husky: 1, Great Pyrenees/Standard Poodle: 1, Greater Swiss Mountain Dog Mix:
4, Greyhound: 7, Greyhound Mix: 8, Greyhound/Dalmatian: 1, Greyhound/Great Da
ne: 1, Greyhound/Labrador Retriever: 1, Harrier: 2, Harrier Mix: 15, Harrier/
Catahoula: 2, Harrier/Labrador Retriever: 1, Harrier/Pointer: 1, Havana Brown
Mix: 1, Havanese: 1, Havanese Mix: 10, Himalayan: 3, Himalayan Mix: 15, Hovaw
art Mix: 1, Ibizan Hound Mix: 2, Ibizan Hound/Pit Bull: 1, Irish Setter/Pit B
ull: 3, Irish Terrier Mix: 5, Irish Terrier/Labrador Retriever: 1, Irish Wolf
hound Mix: 3, Irish Wolfhound/Australian Shepherd: 1, Irish Wolfhound/Great D
ane: 1, Irish Wolfhound/Great Pyrenees: 1, Italian Greyhound: 1, Italian Grey
hound Mix: 12, Italian Greyhound/Beagle: 1, Italian Greyhound/Chihuahua Short
hair: 1, Jack Russell Terrier: 16, Jack Russell Terrier Mix: 146, Jack Russel
l Terrier/American Staffordshire Terrier: 1, Jack Russell Terrier/Australian
Cattle Dog: 6, Jack Russell Terrier/Basenji: 1, Jack Russell Terrier/Basset H
ound: 1, Jack Russell Terrier/Beagle: 4, Jack Russell Terrier/Border Collie:
1, Jack Russell Terrier/Boston Terrier: 1, Jack Russell Terrier/Cairn Terrie
r: 2, Jack Russell Terrier/Cardigan Welsh Corgi: 1, Jack Russell Terrier/Cava
lier Span: 1, Jack Russell Terrier/Chihuahua Shorthair: 12, Jack Russell Terr
ier/Dachshund: 5, Jack Russell Terrier/Dachshund Wirehair: 1, Jack Russell Te
rrier/Italian Greyhound: 1, Jack Russell Terrier/Labrador Retriever: 2, Jack
Russell Terrier/Miniature Poodle: 2, Jack Russell Terrier/Miniature Schnauze
r: 1, Jack Russell Terrier/Papillon: 1, Jack Russell Terrier/Pit Bull: 5, Jac
k Russell Terrier/Pointer: 1, Jack Russell Terrier/Pug: 2, Jack Russell Terri
er/Rat Terrier: 2, Jack Russell Terrier/Shih Tzu: 3, Jack Russell Terrier/Unk
nown: 1, Jack Russell Terrier/Welsh Terrier: 1, Jack Russell Terrier/Whippet:
1, Japanese Bobtail Mix: 6, Japanese Chin: 1, Japanese Chin Mix: 2, Javanese

Mix: 2, Jindo Mix: 3, Keeshond: 1, Keeshond Mix: 5, Kuvasz/Labrador Retrieve
r: 1, Labrador Retriever: 69, Labrador Retriever Mix: 1363, Labrador Retrieve
r/Alaskan Husky: 1, Labrador Retriever/American Bulldog: 2, Labrador Retrieve
r/American Pit Bull Terrier: 1, Labrador Retriever/American Staffordshire Ter
rier: 1, Labrador Retriever/Anatol Shepherd: 12, Labrador Retriever/Australia
n Cattle Dog: 29, Labrador Retriever/Australian Kelpie: 3, Labrador Retrieve
r/Australian Shepherd: 17, Labrador Retriever/Basset Hound: 9, Labrador Retri
ever/Beagle: 14, Labrador Retriever/Beauceron: 1, Labrador Retriever/Black Mo
uth Cur: 3, Labrador Retriever/Black/Tan Hound: 4, Labrador Retriever/Blue La
cy: 8, Labrador Retriever/Border Collie: 37, Labrador Retriever/Border Terrie
r: 2, Labrador Retriever/Boston Terrier: 1, Labrador Retriever/Boxer: 21, Lab
rador Retriever/Catahoula: 11, Labrador Retriever/Chesa Bay Retr: 1, Labrador
Retriever/Chinese Sharpei: 10, Labrador Retriever/Chow Chow: 20, Labrador Ret
riever/Cocker Spaniel: 2, Labrador Retriever/Collie Rough: 2, Labrador Retrie
ver/Collie Smooth: 2, Labrador Retriever/Dachshund: 3, Labrador Retriever/Dal
matian: 2, Labrador Retriever/Doberman Pinsch: 4, Labrador Retriever/English
Pointer: 1, Labrador Retriever/Flat Coat Retriever: 4, Labrador Retriever/Ger
man Shepherd: 56, Labrador Retriever/Golden Retriever: 5, Labrador Retriever/
Great Dane: 6, Labrador Retriever/Great Pyrenees: 19, Labrador Retriever/Grey
hound: 3, Labrador Retriever/Harrier: 1, Labrador Retriever/Jack Russell Terr
ier: 3, Labrador Retriever/Labrador Retriever: 1, Labrador Retriever/Mastiff:
1, Labrador Retriever/Miniature Poodle: 2, Labrador Retriever/Newfoundland:
2, Labrador Retriever/Pembroke Welsh Corgi: 2, Labrador Retriever/Pit Bull: 8
1, Labrador Retriever/Plott Hound: 15, Labrador Retriever/Pointer: 18, Labrad
or Retriever/Queensland Heeler: 3, Labrador Retriever/Redbone Hound: 2, Labra
dor Retriever/Rhod Ridgeback: 6, Labrador Retriever/Rottweiler: 10, Labrador
Retriever/Siberian Husky: 6, Labrador Retriever/Spanish Water Dog: 1, Labrado
r Retriever/St. Bernard Rough Coat: 1, Labrador Retriever/Staffordshire: 4, L
abrador Retriever/Standard Poodle: 3, Labrador Retriever/Standard Schnauzer:
1, Labrador Retriever/Vizsla: 2, Labrador Retriever/Weimaraner: 1, Landseer M
ix: 6, Leonberger: 1, Leonberger Mix: 8, Lhasa Apso: 11, Lhasa Apso Mix: 41,
Lhasa Apso/Cairn Terrier: 1, Lhasa Apso/Cocker Spaniel: 1, Lhasa Apso/Dachshu
nd: 1, Lhasa Apso/Havanese: 1, Lhasa Apso/Miniature Poodle: 6, Lhasa Apso/Min
iature Schnauzer: 2, Lhasa Apso/Pbgv: 1, Lhasa Apso/Shih Tzu: 3, Lhasa Apso/W
est Highland: 1, Lhasa Apso/Yorkshire Terrier: 1, Lowchen Mix: 1, Maine Coon:
3, Maine Coon Mix: 44, Maltese: 19, Maltese Mix: 49, Maltese/Border Terrier:
1, Maltese/Chihuahua Shorthair: 1, Maltese/Cocker Spaniel: 1, Maltese/Dachshu
nd: 1, Maltese/Jack Russell Terrier: 1, Maltese/Miniature Poodle: 29, Maltes
e/Papillon: 1, Maltese/Shih Tzu: 2, Maltese/Toy Poodle: 3, Maltese/Yorkshire:
1, Maltese/Yorkshire Terrier: 2, Manchester Terrier Mix: 40, Manchester Terri
er/Australian Kelpie: 2, Manchester Terrier/Beagle: 4, Manchester Terrier/Chi
huahua Shorthair: 2, Manchester Terrier/Dachshund: 4, Manchester Terrier/Jack
Russell Terrier: 2, Manchester Terrier/Norfolk Terrier: 1, Manchester Terrie
r/Toy Fox Terrier: 1, Manx: 1, Manx Mix: 44, Manx/Domestic Longhair: 1, Manx/
Domestic Shorthair: 2, Mastiff Mix: 23, Mastiff/Boxer: 1, Mastiff/Chinese Sha
rpei: 7, Mastiff/Labrador Retriever: 1, Mastiff/Pit Bull: 1, Mastiff/Rottweil
er: 3, Mexican Hairless Mix: 1, Miniature Pinscher: 15, Miniature Pinscher Mi
x: 69, Miniature Pinscher/Australian Cattle Dog: 1, Miniature Pinscher/Chihua
hua Shorthair: 15, Miniature Pinscher/Maltese: 1, Miniature Pinscher/Pomerani
an: 1, Miniature Poodle: 21, Miniature Poodle Mix: 233, Miniature Poodle/Aust
ralian Cattle Dog: 1, Miniature Poodle/Bichon Frise: 2, Miniature Poodle/Chih
uahua Shorthair: 2, Miniature Poodle/Cocker Spaniel: 5, Miniature Poodle/Dach
shund: 4, Miniature Poodle/English Cocker Spaniel: 1, Miniature Poodle/Golden
Retriever: 1, Miniature Poodle/Havanese: 1, Miniature Poodle/Italian Greyhoun
d: 1, Miniature Poodle/Labrador Retriever: 2, Miniature Poodle/Lhasa Apso: 2,
Miniature Poodle/Maltese: 9, Miniature Poodle/Miniature Schnauzer: 19, Miniat
ure Poodle/Pomeranian: 1, Miniature Poodle/Shih Tzu: 1, Miniature Poodle/York

shire Terrier: 4, Miniature Schnauzer: 22, Miniature Schnauzer Mix: 136, Mini
ature Schnauzer/Australian Cattle Dog: 1, Miniature Schnauzer/Cairn Terrier:
2, Miniature Schnauzer/Chihuahua Shorthair: 3, Miniature Schnauzer/Jack Russe
ll Terrier: 1, Miniature Schnauzer/Labrador Retriever: 1, Miniature Schnauze
r/Lhasa Apso: 1, Miniature Schnauzer/Maltese: 1, Miniature Schnauzer/Miniatur
e Poodle: 41, Miniature Schnauzer/Queensland Heeler: 1, Miniature Schnauzer/S
cottish Terrier: 1, Miniature Schnauzer/Shih Tzu: 1, Miniature Schnauzer/Soft
Coated Wheaten Terrier: 1, Miniature Schnauzer/West Highland: 1, Miniature Sc
hnauzer/Whippet: 1, Miniature Schnauzer/Yorkshire Terrier: 5, Munchkin Longha
ir Mix: 1, Neapolitan Mastiff Mix: 2, Newfoundland Mix: 6, Newfoundland/Austr
alian Cattle Dog: 1, Newfoundland/Border Collie: 1, Newfoundland/Great Pyrene
es: 1, Newfoundland/Labrador Retriever: 2, Newfoundland/Queensland Heeler: 1,
Norfolk Terrier: 1, Norfolk Terrier Mix: 21, Norfolk Terrier/Border Terrier:
1, Norfolk Terrier/Cairn Terrier: 1, Norfolk Terrier/Chihuahua Shorthair: 1,
Norfolk Terrier/Dachshund: 1, Norfolk Terrier/Dachshund Wirehair: 1, Norfolk
Terrier/Yorkshire Terrier: 3, Norwegian Elkhound Mix: 1, Norwegian Forest Cat
Mix: 1, Norwich Terrier Mix: 15, Norwich Terrier/Border Terrier: 1, Norwich T
errier/Cairn Terrier: 1, Norwich Terrier/Pug: 1, Nova Scotia Duck Tolling Ret
riever: 1, Nova Scotia Duck Tolling Retriever Mix: 1, Nova Scotia Duck Tollin
g Retriever/Golden Retriever: 1, Ocicat Mix: 1, Old English Bulldog Mix: 3, O
ld English Sheepdog: 2, Otterhound Mix: 1, Papillon: 1, Papillon Mix: 13, Pap
illon/Australian Cattle Dog: 1, Papillon/Border Collie: 1, Papillon/Chihuahua
Shorthair: 1, Papillon/Pomeranian: 1, Papillon/Skye Terrier: 1, Parson Russel
l Terrier: 4, Parson Russell Terrier Mix: 18, Parson Russell Terrier/Chihuahu
a Shorthair: 1, Patterdale Terr Mix: 5, Patterdale Terr/Chihuahua Shorthair:
1, Pbgv: 6, Pbgv Mix: 7, Pekingese: 5, Pekingese Mix: 26, Pekingese/Cavalier
Span: 1, Pekingese/Chihuahua Shorthair: 1, Pekingese/Dachshund: 1, Pekingese/
Jack Russell Terrier: 1, Pekingese/Lhasa Apso: 1, Pekingese/Miniature Poodle:
1, Pekingese/Pug: 1, Pembroke Welsh Corgi: 4, Pembroke Welsh Corgi Mix: 18, P
embroke Welsh Corgi/Australian Shepherd: 1, Pembroke Welsh Corgi/Basset Houn
d: 1, Pembroke Welsh Corgi/Brittany: 1, Pembroke Welsh Corgi/Chihuahua Shorth
air: 1, Pembroke Welsh Corgi/Dachshund: 1, Pembroke Welsh Corgi/German Shephe
rd: 1, Pembroke Welsh Corgi/Golden Retriever: 1, Pembroke Welsh Corgi/Pit Bul
l: 1, Persian: 2, Persian Mix: 12, Pharaoh Hound: 2, Pharaoh Hound Mix: 9, Ph
araoh Hound/Australian Cattle Dog: 1, Pharaoh Hound/Border Collie: 1, Picardy
Sheepdog Mix: 3, Pit Bull: 66, Pit Bull Mix: 1906, Pit Bull/American Bulldog:
1, Pit Bull/American Foxhound: 1, Pit Bull/Australian Cattle Dog: 9, Pit Bul
l/Basset Hound: 4, Pit Bull/Beagle: 2, Pit Bull/Black Mouth Cur: 1, Pit Bull/
Blue Lacy: 1, Pit Bull/Border Collie: 1, Pit Bull/Border Terrier: 1, Pit Bul
l/Boston Terrier: 2, Pit Bull/Boxer: 18, Pit Bull/Bull Terrier: 1, Pit Bull/B
ulldog: 2, Pit Bull/Cardigan Welsh Corgi: 1, Pit Bull/Catahoula: 8, Pit Bull/
Chihuahua Shorthair: 6, Pit Bull/Chinese Sharpei: 12, Pit Bull/Chow Chow: 1,
Pit Bull/Dachshund: 1, Pit Bull/Dalmatian: 2, Pit Bull/Dogue De Bordeaux: 1,
Pit Bull/English Bulldog: 2, Pit Bull/Flat Coat Retriever: 2, Pit Bull/German
Shepherd: 5, Pit Bull/Labrador Retriever: 30, Pit Bull/Mastiff: 3, Pit Bull/P
it Bull: 2, Pit Bull/Plott Hound: 4, Pit Bull/Pointer: 7, Pit Bull/Pug: 1, Pi
t Bull/Queensland Heeler: 1, Pit Bull/Rhod Ridgeback: 1, Pit Bull/Rottweiler:
2, Pit Bull/Siberian Husky: 2, Pit Bull/St. Bernard Smooth Coat: 1, Pit Bull/
Weimaraner: 2, Pixiebob Shorthair Mix: 3, Plott Hound: 4, Plott Hound Mix: 7
3, Plott Hound/Australian Cattle Dog: 1, Plott Hound/Black/Tan Hound: 1, Plot
t Hound/Blue Lacy: 1, Plott Hound/Boxer: 2, Plott Hound/Carolina Dog: 1, Plot
t Hound/Labrador Retriever: 10, Plott Hound/Pit Bull: 2, Plott Hound/Rhod Rid
geback: 1, Podengo Pequeno Mix: 5, Pointer: 3, Pointer Mix: 93, Pointer/Ameri
can Pit Bull Terrier: 3, Pointer/Anatol Shepherd: 1, Pointer/Australian Cattl
e Dog: 3, Pointer/Blue Lacy: 1, Pointer/Border Collie: 3, Pointer/Boxer: 2, P
ointer/Brittany: 1, Pointer/Bull Terrier: 1, Pointer/Catahoula: 4, Pointer/Ch
inese Sharpei: 1, Pointer/Collie Smooth: 1, Pointer/Harrier: 2, Pointer/Jack

Russell Terrier: 1, Pointer/Labrador Retriever: 12, Pointer/Pit Bull: 4, Poin
ter/Queensland Heeler: 1, Pointer/Rhod Ridgeback: 1, Pointer/Staffordshire:
1, Pomeranian: 9, Pomeranian Mix: 39, Pomeranian/Border Terrier: 1, Pomerania
n/Chihuahua Longhair: 4, Pomeranian/Chihuahua Shorthair: 2, Pomeranian/Jack R
ussell Terrier: 1, Pomeranian/Schipperke: 1, Port Water Dog Mix: 2, Presa Can
ario: 1, Presa Canario Mix: 2, Pug: 25, Pug Mix: 49, Pug/Beagle: 11, Pug/Bord
er Terrier: 1, Pug/Chihuahua Shorthair: 17, Pug/Italian Greyhound: 1, Pug/Lab
rador Retriever: 2, Pug/Pekingese: 1, Pug/Pit Bull: 1, Pug/Pomeranian: 1, Pu
g/Schipperke: 1, Pug/Staffordshire: 1, Queensland Heeler: 1, Queensland Heele
r Mix: 51, Queensland Heeler/Basset Hound: 1, Queensland Heeler/Border Colli
e: 1, Queensland Heeler/Chihuahua Shorthair: 1, Queensland Heeler/Dachshund:
1, Queensland Heeler/Great Dane: 1, Queensland Heeler/Pit Bull: 2, Queensland
Heeler/Pointer: 1, Queensland Heeler/Shetland Sheepdog: 1, Ragdoll: 1, Ragdol
l Mix: 11, Rat Terrier: 13, Rat Terrier Mix: 157, Rat Terrier/Australian Catt
le Dog: 2, Rat Terrier/Basenji: 1, Rat Terrier/Beagle: 4, Rat Terrier/Boston
Terrier: 3, Rat Terrier/Cardigan Welsh Corgi: 3, Rat Terrier/Chihuahua Longha
ir: 1, Rat Terrier/Chihuahua Shorthair: 22, Rat Terrier/Dachshund: 2, Rat Ter
rier/Jack Russell Terrier: 1, Rat Terrier/Miniature Poodle: 2, Rat Terrier/Pi
t Bull: 1, Rat Terrier/Pointer: 1, Rat Terrier/Pug: 1, Rat Terrier/Queensland
Heeler: 1, Redbone Hound: 1, Redbone Hound Mix: 21, Redbone Hound/Anatol Shep
herd: 1, Redbone Hound/Labrador Retriever: 1, Redbone Hound/Redbone Hound: 1,
Rhod Ridgeback: 4, Rhod Ridgeback Mix: 25, Rhod Ridgeback/German Shepherd: 1,
Rhod Ridgeback/Labrador Retriever: 5, Rhod Ridgeback/Pit Bull: 2, Rhod Ridgeb
ack/Plott Hound: 1, Rottweiler: 39, Rottweiler Mix: 113, Rottweiler/Australia
n Cattle Dog: 3, Rottweiler/Basset Hound: 1, Rottweiler/Beagle: 1, Rottweile
r/Chow Chow: 3, Rottweiler/German Shepherd: 9, Rottweiler/Labrador Retriever:
10, Rottweiler/Pit Bull: 1, Rottweiler/Rhod Ridgeback: 1, Russian Blue: 2, Ru
ssian Blue Mix: 33, Saluki: 1, Saluki Mix: 3, Saluki/Labrador Retriever: 1, S
amoyed Mix: 3, Schipperke Mix: 7, Schipperke/Chihuahua Shorthair: 2, Schnauze
r Giant Mix: 4, Scottish Terrier Mix: 2, Scottish Terrier/Basset Hound: 1, Sc
ottish Terrier/Cairn Terrier: 1, Scottish Terrier/Miniature Poodle: 1, Sealyh
am Terr Mix: 1, Shetland Sheepdog: 5, Shetland Sheepdog Mix: 22, Shetland She
epdog/Basenji: 1, Shetland Sheepdog/Border Collie: 1, Shetland Sheepdog/Britt
any: 1, Shetland Sheepdog/Cardigan Welsh Corgi: 1, Shetland Sheepdog/Cavalier
Span: 1, Shetland Sheepdog/Chihuahua Longhair: 2, Shetland Sheepdog/Dachshund
Longhair: 1, Shetland Sheepdog/Keeshond: 1, Shetland Sheepdog/Pomeranian: 1,
Shiba Inu: 2, Shiba Inu Mix: 17, Shiba Inu/Dachshund: 1, Shiba Inu/Siberian H
usky: 1, Shih Tzu: 53, Shih Tzu Mix: 100, Shih Tzu/Affenpinscher: 1, Shih Tz
u/Australian Shepherd: 1, Shih Tzu/Cardigan Welsh Corgi: 1, Shih Tzu/Cavalier
Span: 1, Shih Tzu/Cocker Spaniel: 2, Shih Tzu/Dachshund: 1, Shih Tzu/Havanes
e: 3, Shih Tzu/Maltese: 2, Shih Tzu/Miniature Poodle: 6, Shih Tzu/Miniature S
chnauzer: 2, Shih Tzu/Norfolk Terrier: 1, Shih Tzu/Pekingese: 1, Shih Tzu/Yor
kshire Terrier: 1, Siamese: 24, Siamese Mix: 389, Siamese/Domestic Shorthair:
11, Siamese/Japanese Bobtail: 2, Siberian Husky: 30, Siberian Husky Mix: 138,
Siberian Husky/Anatol Shepherd: 1, Siberian Husky/Australian Cattle Dog: 2, S
iberian Husky/Australian Shepherd: 1, Siberian Husky/Beagle: 1, Siberian Husk
y/Border Collie: 2, Siberian Husky/Catahoula: 1, Siberian Husky/German Shephe
rd: 11, Siberian Husky/Labrador Retriever: 5, Siberian Husky/Pit Bull: 2, Sil
ky Terrier Mix: 7, Silky Terrier/Yorkshire Terrier: 1, Skye Terrier Mix: 3, S
kye Terrier/Miniature Poodle: 1, Smooth Fox Terrier Mix: 10, Snowshoe: 2, Sno
wshoe Mix: 75, Snowshoe/Ragdoll: 1, Soft Coated Wheaten Terrier: 3, Soft Coat
ed Wheaten Terrier Mix: 23, Soft Coated Wheaten Terrier/Labrador Retriever:
1, Soft Coated Wheaten Terrier/Standard Poodle: 1, Soft Coated Wheaten Terrie
r/Yorkshire Terrier: 1, Spanish Mastiff Mix: 1, Sphynx: 2, Spinone Italiano M
ix: 1, St. Bernard Rough Coat: 1, St. Bernard Rough Coat Mix: 5, St. Bernard
Rough Coat/Border Collie: 1, St. Bernard Rough Coat/German Shepherd: 1, St. B
ernard Rough Coat/Labrador Retriever: 1, St. Bernard Rough Coat/Standard Schn

```
auzer: 1, St. Bernard Smooth Coat: 1, St. Bernard Smooth Coat Mix: 10, St. Be
rnard Smooth Coat/Anatol Shepherd: 1, Staffordshire: 8, Staffordshire Mix: 9
4, Staffordshire/Beagle: 1, Staffordshire/Blue Lacy: 1, Staffordshire/Boston
Terrier: 2, Staffordshire/Boxer: 1, Staffordshire/Bull Terrier: 2, Staffordsh
ire/English Bulldog: 1, Staffordshire/French Bulldog: 1, Staffordshire/Labrad
or Retriever: 1, Standard Poodle: 4, Standard Poodle Mix: 11, Standard Poodl
e/Labrador Retriever: 4, Standard Poodle/Whippet: 1, Standard Schnauzer: 3, S
tandard Schnauzer Mix: 12, Standard Schnauzer/Border Terrier: 1, Standard Sch
nauzer/Toy Poodle: 1, Swedish Vallhund Mix: 5, Swedish Vallhund/Miniature Poo
dle: 1, Swiss Hound/Pit Bull: 1, Tibetan Spaniel: 1, Tibetan Spaniel Mix: 3,
Tibetan Terrier Mix: 6, Tibetan Terrier/Lhasa Apso: 2, Tonkinese Mix: 4, Toy
Fox Terrier Mix: 7, Toy Fox Terrier/Dachshund: 1, Toy Poodle: 7, Toy Poodle M
ix: 36, Toy Poodle/Maltese: 1, Toy Poodle/Yorkshire Terrier: 1, Treeing Cur M
ix: 4, Treeing Tennesse Brindle: 1, Treeing Walker Coonhound Mix: 8, Treeing
Walker Coonhound/Labrador Retriever: 1, Turkish Van Mix: 2, Unknown Mix: 2, V
izsla: 4, Vizsla Mix: 7, Vizsla/Boxer: 1, Vizsla/Catahoula: 1, Vizsla/Golden
Retriever: 2, Vizsla/Rhod Ridgeback: 1, Vizsla/Staffordshire: 1, Weimaraner:
4, Weimaraner Mix: 19, Weimaraner/Labrador Retriever: 3, Weimaraner/Pit Bull:
1, Welsh Springer Spaniel: 1, Welsh Springer Spaniel Mix: 1, Welsh Terrier Mi
x: 5, West Highland: 5, West Highland Mix: 14, West Highland/Cairn Terrier:
2, West Highland/Chihuahua Shorthair: 1, Whippet Mix: 21, Whippet/Australian
Cattle Dog: 1, Whippet/Australian Kelpie: 1, Whippet/Beagle: 1, Whippet/Borzo
i: 1, Whippet/Chihuahua Shorthair: 1, Whippet/Dachshund: 2, Whippet/Jack Russ
ell Terrier: 1, Whippet/Labrador Retriever: 2, Whippet/Miniature Pinscher: 1,
Whippet/Shetland Sheepdog: 1, Whippet/Whippet: 1, Wire Hair Fox Terrier: 2, W
ire Hair Fox Terrier Mix: 12, Wire Hair Fox Terrier/Beagle: 2, Wire Hair Fox
Terrier/Chihuahua Shorthair: 4, Wire Hair Fox Terrier/German Shepherd: 1, Wir
ehaired Pointing Griffon Mix: 2, Yorkshire Terrier: 24, Yorkshire Terrier Mi
x: 143, Yorkshire Terrier/Border Terrier: 3, Yorkshire Terrier/Bruss Griffon:
1, Yorkshire Terrier/Cairn Terrier: 6, Yorkshire Terrier/Chihuahua Longhair:
7, Yorkshire Terrier/Chihuahua Shorthair: 9, Yorkshire Terrier/Dachshund Wire
hair: 1, Yorkshire Terrier/Maltese: 4, Yorkshire Terrier/Miniature Poodle: 1
2, Yorkshire Terrier/Miniature Schnauzer: 8, Yorkshire Terrier/Norfolk Terrie
r: 1, Yorkshire Terrier/Parson Russell Terrier: 1, Yorkshire Terrier/Pomerani
an: 1, Yorkshire Terrier/Rat Terrier: 4, Yorkshire Terrier/Toy Poodle: 1,
```

In [21]:
```
# Remove the secondary breed name.
def split_breed(x):
    if '/' in x['Breed'] : return x['Breed'].split('/', 1)[1]
    else : return x['Breed']

train_df['Breed'] = train_df.apply(split_breed, axis=1)
```

In [22]:
```
# Remove the word "mix" from the breed type.
train_df['Breed'] = train_df['Breed'].str.replace(' Mix', '')
```

In [23]:
```
print("We have taken the unique number of breeds from 1,380 to",
      train_df['Breed'].describe()[1], "by removing the secondary breed type",
      "and the word \"mix\".")
```

```
We have taken the unique number of breeds from 1,380 to 225 by removing the s
econdary breed type and the word "mix".
```

In [24]:

```
# Evaluate the Color feature set for combination.
train_df_color_list = train_df['Color'].tolist()
train_df_color_dict = {i:train_df_color_list.count(i) for i in set(train_df_co
lor_list)}

print("Number of unique colors within the dataset:", train_df['Color'].describ
e()[1])

# Analyzing the different combinations of colors.
for k in sorted(train_df_color_dict):
    print(k,": ",train_df_color_dict[k], sep="", end=", ")
```

Number of unique colors within the dataset: 366
Agouti: 1, Agouti/Brown Tabby: 1, Apricot: 21, Apricot/Brown: 2, Apricot/Whit
e: 3, Black: 2292, Black Brindle: 20, Black Brindle/Black: 1, Black Brindle/B
rown: 3, Black Brindle/Brown Brindle: 1, Black Brindle/White: 74, Black Smok
e: 46, Black Smoke/Brown Tabby: 1, Black Smoke/White: 15, Black Tabby: 42, Bl
ack Tabby/Orange: 1, Black Tabby/White: 18, Black Tiger/White: 2, Black/Blac
k: 4, Black/Black Brindle: 5, Black/Black Smoke: 3, Black/Black Tabby: 1, Bla
ck/Blue: 3, Black/Blue Merle: 8, Black/Blue Tick: 6, Black/Brown: 436, Black/
Brown Brindle: 45, Black/Brown Merle: 2, Black/Buff: 5, Black/Chocolate: 2, B
lack/Cream: 10, Black/Gray: 38, Black/Orange: 3, Black/Red: 17, Black/Silver:
3, Black/Silver Tabby: 1, Black/Tan: 672, Black/Tricolor: 39, Black/White: 28
24, Black/Yellow: 1, Black/Yellow Brindle: 2, Blue: 450, Blue Cream: 11, Blue
Cream/Blue Tabby: 1, Blue Cream/Blue Tiger: 1, Blue Cream/Buff: 1, Blue Crea
m/Tortie: 1, Blue Cream/White: 4, Blue Merle: 77, Blue Merle/Black: 4, Blue M
erle/Blue Merle: 1, Blue Merle/Brown: 7, Blue Merle/Cream: 1, Blue Merle/Gra
y: 1, Blue Merle/Red: 1, Blue Merle/Red Merle: 1, Blue Merle/Tan: 26, Blue Me
rle/Tricolor: 2, Blue Merle/White: 44, Blue Point: 29, Blue Point/White: 2, B
lue Smoke: 4, Blue Smoke/Brown: 1, Blue Tabby: 433, Blue Tabby/Blue Cream: 1,
Blue Tabby/Cream: 1, Blue Tabby/Orange: 1, Blue Tabby/Tan: 1, Blue Tabby/Whit
e: 241, Blue Tick: 18, Blue Tick/Black: 11, Blue Tick/Brown: 3, Blue Tick/Bro
wn Brindle: 1, Blue Tick/Red: 1, Blue Tick/Red Tick: 3, Blue Tick/Tan: 4, Blu
e Tick/White: 3, Blue Tiger: 1, Blue Tiger/White: 7, Blue/Black: 4, Blue/Brow
n: 1, Blue/Brown Brindle: 3, Blue/Cream: 3, Blue/Orange: 1, Blue/Tan: 33, Blu
e/Tortie: 1, Blue/White: 702, Blue/Yellow Brindle: 1, Brown: 639, Brown Brind
le: 232, Brown Brindle/Black: 7, Brown Brindle/Blue: 1, Brown Brindle/Blue Cr
eam: 1, Brown Brindle/Blue Tick: 1, Brown Brindle/Brown Brindle: 1, Brown Bri
ndle/Brown Merle: 1, Brown Brindle/Red Tick: 3, Brown Brindle/Tan: 2, Brown B
rindle/White: 450, Brown Merle: 35, Brown Merle/Black: 2, Brown Merle/Blue Me
rle: 1, Brown Merle/Tan: 1, Brown Merle/White: 33, Brown Tabby: 1635, Brown T
abby/Black: 9, Brown Tabby/Brown: 1, Brown Tabby/Gray: 2, Brown Tabby/Gray Ta
bby: 1, Brown Tabby/Orange: 2, Brown Tabby/Tortie: 2, Brown Tabby/White: 940,
Brown Tiger: 3, Brown Tiger/White: 1, Brown/Black: 333, Brown/Black Brindle:
1, Brown/Black Tabby: 1, Brown/Blue: 1, Brown/Blue Smoke: 1, Brown/Brown: 2,
Brown/Brown Merle: 1, Brown/Buff: 2, Brown/Chocolate: 2, Brown/Cream: 4, Brow
n/Gray: 8, Brown/Red: 6, Brown/Red Tick: 2, Brown/Silver: 1, Brown/Tan: 51, B
rown/Tricolor: 12, Brown/White: 884, Buff: 199, Buff/Black: 8, Buff/Brown: 1,
Buff/Gray: 2, Buff/Red: 1, Buff/Tan: 11, Buff/White: 43, Buff/Yellow: 2, Cali
co: 517, Calico Point: 25, Calico Point/White: 2, Calico/Black: 2, Calico/Blu
e Cream: 1, Calico/Blue Tabby: 1, Calico/Brown: 1, Calico/Brown Tabby: 3, Cal
ico/Orange Tabby: 1, Calico/Tricolor: 2, Calico/White: 24, Chocolate: 137, Ch
ocolate Point: 16, Chocolate Point/White: 5, Chocolate/Black: 2, Chocolate/Bl
ack Smoke: 2, Chocolate/Brown: 6, Chocolate/Brown Brindle: 1, Chocolate/Brown
Merle: 1, Chocolate/Cream: 1, Chocolate/Gold: 1, Chocolate/Gray: 1, Chocolat
e/Red: 3, Chocolate/Red Tick: 1, Chocolate/Tan: 66, Chocolate/Tricolor: 2, Ch
ocolate/White: 224, Cream: 151, Cream Tabby: 198, Cream Tabby/White: 79, Crea
m/Black: 16, Cream/Brown: 2, Cream/Gray: 2, Cream/Orange: 1, Cream/Red: 1, Cr
eam/Red Tick: 1, Cream/Seal Point: 1, Cream/Tan: 8, Cream/White: 38, Fawn: 7
3, Fawn/Black: 19, Fawn/Blue: 3, Fawn/Brown: 1, Fawn/Brown Brindle: 1, Fawn/T
an: 2, Fawn/Tricolor: 1, Fawn/White: 109, Flame Point: 85, Gold: 52, Gold/Bla
ck: 1, Gold/Buff: 1, Gold/Gold: 2, Gold/Tan: 1, Gold/White: 19, Gold/Yellow:
1, Gray: 94, Gray Tabby: 35, Gray Tabby/Black: 1, Gray Tabby/White: 15, Gray/
Black: 16, Gray/Brown: 3, Gray/Cream: 2, Gray/Red: 1, Gray/Silver: 4, Gray/Ta
n: 11, Gray/White: 96, Lilac Point: 39, Liver Tick: 3, Liver Tick/White: 1, L
iver/Buff: 1, Liver/Tan: 3, Liver/White: 11, Lynx Point: 168, Lynx Point/Brow
n Tabby: 1, Lynx Point/Gray Tabby: 1, Lynx Point/White: 13, Orange: 15, Orang
e Tabby: 841, Orange Tabby/Apricot: 1, Orange Tabby/Black: 1, Orange Tabby/Or
ange: 1, Orange Tabby/White: 455, Orange Tiger: 1, Orange/Orange Tabby: 1, Or

```
ange/Tan: 1, Orange/White: 26, Pink: 1, Red: 337, Red Merle: 26, Red Merle/Bl
ack: 2, Red Merle/Tan: 1, Red Merle/White: 23, Red Tick: 17, Red Tick/Black:
1, Red Tick/Blue Tick: 1, Red Tick/Brown: 1, Red Tick/Brown Merle: 1, Red Tic
k/Red: 2, Red Tick/Tan: 2, Red Tick/White: 15, Red/Black: 62, Red/Blue: 1, Re
d/Brown: 8, Red/Buff: 2, Red/Cream: 4, Red/Gold: 1, Red/Red Merle: 2, Red/Red
Tick: 2, Red/Silver: 1, Red/Tan: 26, Red/Tricolor: 2, Red/White: 331, Ruddy/C
ream: 1, Sable: 198, Sable/Black: 12, Sable/Brown: 7, Sable/Buff: 1, Sable/Cr
eam: 3, Sable/Red: 1, Sable/Tan: 16, Sable/White: 86, Seal Point: 136, Seal P
oint/Brown: 3, Seal Point/White: 19, Silver: 17, Silver Lynx Point: 2, Silver
Tabby: 28, Silver Tabby/Black: 1, Silver Tabby/White: 12, Silver/Black: 5, Si
lver/Brown: 4, Silver/Chocolate: 2, Silver/Red: 1, Silver/Tan: 20, Silver/Whi
te: 4, Tan: 628, Tan/Apricot: 1, Tan/Black: 183, Tan/Blue: 4, Tan/Brown: 24,
Tan/Brown Brindle: 1, Tan/Brown Merle: 1, Tan/Buff: 9, Tan/Cream: 9, Tan/Gol
d: 2, Tan/Gray: 16, Tan/Red: 4, Tan/Silver: 11, Tan/Tan: 6, Tan/Tricolor: 1,
Tan/White: 773, Tan/Yellow Brindle: 1, Torbie: 335, Torbie/Blue Cream: 1, Tor
bie/Brown: 1, Torbie/Calico: 1, Torbie/White: 60, Tortie: 530, Tortie Point:
32, Tortie Point/Lynx Point: 1, Tortie Point/White: 1, Tortie/Black: 2, Torti
e/Black Smoke: 1, Tortie/Blue Cream: 9, Tortie/Brown: 1, Tortie/Calico: 1, To
rtie/Orange: 3, Tortie/White: 33, Tricolor: 752, Tricolor/Black: 11, Tricolo
r/Blue: 2, Tricolor/Blue Merle: 1, Tricolor/Brown: 7, Tricolor/Brown Merle:
1, Tricolor/Calico: 1, Tricolor/Chocolate: 2, Tricolor/Silver: 2, Tricolor/Ta
n: 3, Tricolor/White: 18, White: 931, White/Apricot: 2, White/Black: 643, Whi
te/Black Brindle: 9, White/Black Smoke: 1, White/Black Tabby: 4, White/Blue:
131, White/Blue Cream: 2, White/Blue Merle: 6, White/Blue Tabby: 16, White/Bl
ue Tick: 2, White/Brown: 569, White/Brown Brindle: 115, White/Brown Merle: 4,
White/Brown Tabby: 98, White/Brown Tiger: 1, White/Buff: 39, White/Calico: 1,
White/Chocolate: 39, White/Chocolate Point: 3, White/Cream: 36, White/Cream T
abby: 5, White/Flame Point: 1, White/Gold: 3, White/Gray: 55, White/Gray Tabb
y: 2, White/Liver: 12, White/Lynx Point: 2, White/Orange: 30, White/Orange Ta
bby: 52, White/Pink: 3, White/Red: 55, White/Red Merle: 4, White/Red Tick: 7,
White/Silver: 3, White/Tan: 389, White/Tricolor: 51, White/White: 12, White/Y
ellow: 2, White/Yellow Brindle: 4, Yellow: 185, Yellow Brindle: 12, Yellow Br
indle/Blue: 1, Yellow Brindle/Tan: 1, Yellow Brindle/White: 17, Yellow/Black:
1, Yellow/White: 43, Yellow/Yellow: 1,
```

In [25]:
```
# Remove the secondary color name to focus on the principal color.
def split_color(x):
    if '/' in x['Color'] : return x['Color'].split('/', 1)[1]
    else : return x['Color']

train_df['Color'] = train_df.apply(split_color, axis=1)
```

In [26]:
```
print("We have taken the unique number of colors from 366 to",
      train_df['Color'].describe()[1], "by focusing on the primary color",
      "\nand removing the secondary color.")
```

```
We have taken the unique number of colors from 366 to 55 by focusing on the p
rimary color
and removing the secondary color.
```

Moving forward, we are focusing on primary breed and color, working under the assumption that someone adopting an animal is likely to focus on the predominant feature in each of these categories. When a potential adopter sees border collie mix or border collie/labrador retriever, they are likely to process this as "border collie" first and foremost. Colors work the same way: the difference between calico vs. calico/white is negligible.

The remaining colors could be reduced further, as they mix colors and coat types (blue vs. blue tabby), but we determined that these were unique enough to keep separate in a single field.

Train Data Transformations and Binarization

In order to transform our dataset and make it ready for model evaluation, we are going to create dummy variables for all of our continuous variables and binarize all of our features. For our age variable, we are going to normalize all of the data, as the variable is continuous.

In [27]:
```
train_df_binarize = train_df.copy()
train_df_binarize.head()
```

Out[27]:

	AnimalID	Name	DateTime	OutcomeType	OutcomeSubtype	AnimalType	SexuponOutco
0	A671945	Hambone	2014-02-05 18:22:00	Return_to_owner	Return to Owner	Dog	Neutered M
1	A656520	Emily	2013-10-06 12:44:00	Euthanasia	Suffering	Cat	Spayed Fem
2	A686464	Pearce	2015-01-24 12:28:00	Adoption	Foster	Dog	Neutered M
3	A683430	NaN	2014-07-04 19:09:00	Transfer	Partner	Cat	Intact M
4	A667013	NaN	2013-11-08 12:52:00	Transfer	Partner	Dog	Neutered M

In [28]:
```
# Split SexuponOutcome into dummy variables, so that both will only have two o
ptions.
train_df_binarize['State'], train_df_binarize['Gender'] = train_df_binarize['S
exuponOutcome'].str.split(' ', 1).str

train_df_binarize['State'] = train_df_binarize['State'].map({'Neutered': 'Fixe
d',
                                                             'Spayed': 'Fixed'
,
                                                             'Intact': 'Intac
t'})
```

In [29]:
```
# Pick the column(s) to binarize.
train_df_binarize_animaltype = pd.get_dummies(train_df_binarize['AnimalType'])
train_df_binarize_state = pd.get_dummies(train_df_binarize['State'])
train_df_binarize_gender = pd.get_dummies(train_df_binarize['Gender'])

# Only keep one, as we only have two variables.
train_df_binarize_animaltype = train_df_binarize_animaltype['Dog']
train_df_binarize_state = train_df_binarize_state['Fixed']
train_df_binarize_gender = train_df_binarize_gender['Male']

# Combine the state binarization and the original dataset.
train_df_binarize = pd.concat([train_df_binarize, train_df_binarize_animaltype
, train_df_binarize_state, train_df_binarize_gender], axis=1)

# Change the name of the columns that we just concatenated.
train_df_binarize = train_df_binarize.rename(index=str, columns={'Dog': 'Anima
lType_Binarize(Dog=1)',
                                                                 'Fixed': 'Sta
te(Fixed=1)',
                                                                 'Male': 'Gend
er(Male=1)'})
```

In [30]:
```
# Pick the column(s) to binarize.
train_df_binarize_breed = pd.get_dummies(train_df_binarize['Breed'])
train_df_binarize_color = pd.get_dummies(train_df_binarize['Color'])

# Need to keep all of the columns, as we have more than two breeds.
# Combine the breed binarization and the original dataset.
train_df_binarize = pd.concat([train_df_binarize, train_df_binarize_breed, tra
in_df_binarize_color], axis=1)
```

In [31]:
```
# Normalize age upon outcome.
train_df_binarize['AgeuponOutcome_norm'] = train_df_binarize['AgeuponOutcome']
/max(train_df_binarize['AgeuponOutcome'])
```

In [32]:
```
# Fill NaN values with the median value for the column.
train_df_binarize['AgeuponOutcome_norm'].fillna(train_df_binarize['AgeuponOutc
ome_norm'].median(), inplace=True)
```

As discussed above, we decided not to binarize the `Name` variable, as the dataset has over 6,000 unique names. With just over 26,700 rows in the dataset, and names only present in just over 19,000, we believed that the dataset would be too sparse if we created a unique column for each name. The sparsity would lead to the names likely not being very good features in predicting animal outcomes.

In [33]:
```
# Print full list of columns from binarization to check work.
print("Num of columns:", len(list(train_df_binarize.columns.values)),
      "\nand columns:", list(train_df_binarize.columns.values),end="")
```

```
Num of columns: 296
and columns: ['AnimalID', 'Name', 'DateTime', 'OutcomeType', 'OutcomeSubtyp
e', 'AnimalType', 'SexuponOutcome', 'AgeuponOutcome', 'Breed', 'Color', 'Stat
e', 'Gender', 'AnimalType_Binarize(Dog=1)', 'State(Fixed=1)', 'Gender(Male=
1)', 'Abyssinian', 'Affenpinscher', 'Afghan Hound', 'Airedale Terrier', 'Akit
a', 'Alaskan Husky', 'Alaskan Malamute', 'American Bulldog', 'American Eskim
o', 'American Foxhound', 'American Pit Bull Terrier', 'American Shorthair',
'American Staffordshire Terrier', 'Anatol Shepherd', 'Angora', 'Australian Ca
ttle Dog', 'Australian Kelpie', 'Australian Shepherd', 'Australian Terrier',
'Balinese', 'Basenji', 'Basset Hound', 'Beagle', 'Bearded Collie', 'Beaucero
n', 'Bedlington Terr', 'Belgian Malinois', 'Belgian Sheepdog', 'Belgian Tervu
ren', 'Bengal', 'Bernese Mountain Dog', 'Bichon Frise', 'Black Mouth Cur', 'B
lack/Tan Hound', 'Bloodhound', 'Blue Lacy', 'Bluetick Hound', 'Boerboel', 'Bo
mbay', 'Border Collie', 'Border Terrier', 'Borzoi', 'Boston Terrier', 'Boxe
r', 'Boykin Span', 'British Shorthair', 'Brittany', 'Bruss Griffon', 'Bull Te
rrier', 'Bull Terrier Miniature', 'Bulldog', 'Bullmastiff', 'Burmese', 'Cairn
Terrier', 'Canaan Dog', 'Cane Corso', 'Cardigan Welsh Corgi', 'Carolina Dog',
'Catahoula', 'Cavalier Span', 'Chesa Bay Retr', 'Chihuahua Longhair', 'Chihua
hua Shorthair', 'Chinese Crested', 'Chinese Sharpei', 'Chow Chow', 'Cocker Sp
aniel', 'Collie Rough', 'Collie Smooth', 'Cornish Rex', 'Cymric', 'Dachshun
d', 'Dachshund Longhair', 'Dachshund Wirehair', 'Dalmatian', 'Devon Rex', 'Do
berman Pinsch', 'Dogo Argentino', 'Dogue De Bordeaux', 'Domestic Longhair',
'Domestic Medium Hair', 'Domestic Shorthair', 'Dutch Shepherd', 'English Bull
dog', 'English Cocker Spaniel', 'English Coonhound', 'English Foxhound', 'Eng
lish Pointer', 'English Setter', 'English Shepherd', 'English Springer Spanie
l', 'Entlebucher', 'Exotic Shorthair', 'Feist', 'Field Spaniel', 'Finnish Spi
tz', 'Flat Coat Retriever', 'French Bulldog', 'German Pinscher', 'German Shep
herd', 'German Shorthair Pointer', 'German Wirehaired Pointer', 'Glen Of Imaa
l', 'Golden Retriever', 'Great Dane', 'Great Pyrenees', 'Greater Swiss Mounta
in Dog', 'Greyhound', 'Harrier', 'Havana Brown', 'Havanese', 'Himalayan', 'Ho
vawart', 'Ibizan Hound', 'Irish Terrier', 'Irish Wolfhound', 'Italian Greyhou
nd', 'Jack Russell Terrier', 'Japanese Bobtail', 'Japanese Chin', 'Javanese',
'Jindo', 'Keeshond', 'Labrador Retriever', 'Landseer', 'Leonberger', 'Lhasa A
pso', 'Lowchen', 'Maine Coon', 'Maltese', 'Manchester Terrier', 'Manx', 'Mast
iff', 'Mexican Hairless', 'Miniature Pinscher', 'Miniature Poodle', 'Miniatur
e Schnauzer', 'Munchkin Longhair', 'Neapolitan Mastiff', 'Newfoundland', 'Nor
folk Terrier', 'Norwegian Elkhound', 'Norwegian Forest Cat', 'Norwich Terrie
r', 'Nova Scotia Duck Tolling Retriever', 'Ocicat', 'Old English Bulldog', 'O
ld English Sheepdog', 'Otterhound', 'Papillon', 'Parson Russell Terrier', 'Pa
tterdale Terr', 'Pbgv', 'Pekingese', 'Pembroke Welsh Corgi', 'Persian', 'Phar
aoh Hound', 'Picardy Sheepdog', 'Pit Bull', 'Pixiebob Shorthair', 'Plott Houn
d', 'Podengo Pequeno', 'Pointer', 'Pomeranian', 'Port Water Dog', 'Presa Cana
rio', 'Pug', 'Queensland Heeler', 'Ragdoll', 'Rat Terrier', 'Redbone Hound',
'Rex', 'Rhod Ridgeback', 'Rottweiler', 'Russian Blue', 'Saluki', 'Samoyed',
'Schipperke', 'Schnauzer Giant', 'Scottish Terrier', 'Sealyham Terr', 'Shetla
nd Sheepdog', 'Shiba Inu', 'Shih Tzu', 'Siamese', 'Siberian Husky', 'Silky Te
rrier', 'Skye Terrier', 'Smooth Fox Terrier', 'Snowshoe', 'Soft Coated Wheate
n Terrier', 'Spanish Mastiff', 'Spanish Water Dog', 'Sphynx', 'Spinone Italia
no', 'St. Bernard Rough Coat', 'St. Bernard Smooth Coat', 'Staffordshire', 'S
tandard Poodle', 'Standard Schnauzer', 'Swedish Vallhund', 'Tan Hound', 'Tan
Hound/Black Mouth Cur', 'Tan Hound/German Shepherd', 'Tan Hound/Labrador Retr
iever', 'Tibetan Spaniel', 'Tibetan Terrier', 'Tonkinese', 'Toy Fox Terrier',
'Toy Poodle', 'Treeing Cur', 'Treeing Tennesse Brindle', 'Treeing Walker Coon
hound', 'Turkish Van', 'Unknown', 'Vizsla', 'Weimaraner', 'Welsh Springer Spa
niel', 'Welsh Terrier', 'West Highland', 'Whippet', 'Wire Hair Fox Terrier',
'Wirehaired Pointing Griffon', 'Yorkshire', 'Yorkshire Terrier', 'Agouti', 'A
pricot', 'Black', 'Black Brindle', 'Black Smoke', 'Black Tabby', 'Blue', 'Blu
```

```
e Cream', 'Blue Merle', 'Blue Point', 'Blue Smoke', 'Blue Tabby', 'Blue Tic
k', 'Blue Tiger', 'Brown', 'Brown Brindle', 'Brown Merle', 'Brown Tabby', 'Br
own Tiger', 'Buff', 'Calico', 'Calico Point', 'Chocolate', 'Chocolate Point',
'Cream', 'Cream Tabby', 'Fawn', 'Flame Point', 'Gold', 'Gray', 'Gray Tabby',
'Lilac Point', 'Liver', 'Liver Tick', 'Lynx Point', 'Orange', 'Orange Tabby',
'Orange Tiger', 'Pink', 'Red', 'Red Merle', 'Red Tick', 'Sable', 'Seal Poin
t', 'Silver', 'Silver Lynx Point', 'Silver Tabby', 'Tan', 'Torbie', 'Tortie',
'Tortie Point', 'Tricolor', 'White', 'Yellow', 'Yellow Brindle', 'AgeuponOutc
ome_norm']
```

In [34]:
```
# Check sparsity of the training set.
print("Sparsity of the binarized training set:", '%.2f'%(100*((np.product(trai
n_df_binarize.shape) - np.count_nonzero(train_df_binarize))/np.product(train_d
f_binarize.shape))),"%")
```

```
Sparsity of the binarized training set: 94.34 %
```

Despite the feature consolidation, the binarization resulted in a very sparse data set.

In [35]:
```
# Export a CSV record of the dataset.
train_df_binarize.to_csv('data/train_df_binarization.csv')
```

Test Data Transformations and Binarization

Our features have to be consistent between the training and test datasets for prediction purposes. Therefore, we are making the same feature transformations and binarizations of our test dataset.

In [36]:
```
test_df_binarize = test_df.copy()
test_df_binarize.head()
```

Out[36]:

	ID	Name	DateTime	AnimalType	SexuponOutcome	AgeuponOutcome	Breed
0	1	Summer	2015-10-12 12:15:00	Dog	Intact Female	10 months	Labrador Retriever Mix
1	2	Cheyenne	2014-07-26 17:59:00	Dog	Spayed Female	2 years	German Shepherd/Siberian Husky
2	3	Gus	2016-01-13 12:20:00	Cat	Neutered Male	1 year	Domestic Shorthair Mix
3	4	Pongo	2013-12-28 18:12:00	Dog	Intact Male	4 months	Collie Smooth Mix
4	5	Skooter	2015-09-24 17:59:00	Dog	Neutered Male	2 years	Miniature Poodle Mix

In [37]:
```
# Split SexuponOutcome into variables, so that both will only have two options
as we did for the training dataset.
test_df_binarize['State'], test_df_binarize['Gender'] = test_df_binarize['Sexu
ponOutcome'].str.split(' ', 1).str

test_df_binarize['State'] = test_df_binarize['State'].map({'Neutered': 'Fixed'
,
                                                           'Spayed': 'Fixed'
,
                                                           'Intact': 'Intac
t'})
test_df_binarize.head()
```

Out[37]:

	ID	Name	DateTime	AnimalType	SexuponOutcome	AgeuponOutcome	Breed
0	1	Summer	2015-10-12 12:15:00	Dog	Intact Female	10 months	Labrador Retriever Mix
1	2	Cheyenne	2014-07-26 17:59:00	Dog	Spayed Female	2 years	German Shepherd/Siberian Husky
2	3	Gus	2016-01-13 12:20:00	Cat	Neutered Male	1 year	Domestic Shorthair Mix
3	4	Pongo	2013-12-28 18:12:00	Dog	Intact Male	4 months	Collie Smooth Mix
4	5	Skooter	2015-09-24 17:59:00	Dog	Neutered Male	2 years	Miniature Poodle Mix

In [38]:
```
# Map all AgeuponOutcome values to numbers as we did in the test dataset. Mapp
ed to weeks and rounded to the nearest week.
test_df_binarize['AgeuponOutcome'] = test_df_binarize['AgeuponOutcome'].map({
'1 week': 1, '1 weeks': 1, '2 weeks': 2,

'3 weeks': 3, '4 weeks': 4, '5 weeks': 5,

'1 month': 4, '2 months': 9, '3 months': 13,

'4 months': 17, '5 months': 21, '6 months': 26,

'7 months': 30, '8 months': 34, '9 months': 38,

'10 months': 43, '11 months': 47, '1 year': 52,

'2 years': 204, '3 years': 156, '4 years': 208,

'5 years': 260, '6 years': 312, '7 years': 364,

'8 years': 416, '9 years': 468, '10 years': 520,

'11 years': 572, '12 years': 624, '13 years': 676,

'14 years': 728, '15 years': 780, '16 years': 832,

'17 years': 884, '18 years': 936, '20 years': 1040,

'0 years': 26, '1 day': 0, '2 days': 0, '3 days': 0,

'4 days': 0, '5 days': 0, '6 days': 0})
```

In [39]:
```
# Change breed types that we changed in train to accurately match test.
# Remove the secondary breed name.
def split_breed(x):
    if '/' in x['Breed'] : return x['Breed'].split('/', 1)[1]
    else : return x['Breed']

test_df_binarize['Breed'] = test_df_binarize.apply(split_breed, axis=1)
```

In [40]:
```
# Remove the word "mix" from the breed type.
test_df_binarize['Breed'] = test_df_binarize['Breed'].str.replace(' Mix', '')
```

In [41]:
```
# Remove the secondary color name to focus on principal color as we did in tra
ining set.
def split_color(x):
    if '/' in x['Color'] : return x['Color'].split('/', 1)[1]
    else : return x['Color']

test_df_binarize['Color'] = test_df_binarize.apply(split_color, axis=1)
```

In [42]:
```
# Pick the column(s) to binarize.
test_df_binarize_animaltype = pd.get_dummies(test_df_binarize['AnimalType'])
test_df_binarize_state = pd.get_dummies(test_df_binarize['State'])
test_df_binarize_gender = pd.get_dummies(test_df_binarize['Gender'])

# Only keep one, as we only have two variables.
test_df_binarize_animaltype = test_df_binarize_animaltype['Dog']
test_df_binarize_state = test_df_binarize_state['Fixed']
test_df_binarize_gender = test_df_binarize_gender['Male']

# Combine the state binarization and the original dataset.
test_df_binarize = pd.concat([test_df_binarize, test_df_binarize_animaltype, t
est_df_binarize_state, test_df_binarize_gender], axis=1)

# Change the name of the columns that we just concatenated.
test_df_binarize = test_df_binarize.rename(index=str, columns={'Dog': 'AnimalT
ype_Binarize(Dog=1)',
                                                               'Fixed': 'State
(Fixed=1)',
                                                               'Male': 'Gender
(Male=1)'})
```

In [43]:
```
# Pick the column(s) to binarize.
test_df_binarize_breed = pd.get_dummies(test_df_binarize['Breed'])
test_df_binarize_color = pd.get_dummies(test_df_binarize['Color'])

# Need to keep all of the columns, as we have more than two breeds.
# Combine the breed binarization and the original dataset.
test_df_binarize = pd.concat([test_df_binarize, test_df_binarize_breed, test_d
f_binarize_color], axis=1)

# Normalize age upon outcome on training set maximium.
test_df_binarize['AgeuponOutcome_norm'] = test_df_binarize['AgeuponOutcome']/m
ax(train_df_binarize['AgeuponOutcome'])
test_df_binarize['AgeuponOutcome_norm'].fillna(test_df_binarize['AgeuponOutcom
e_norm'].median(), inplace=True)
```

In [44]:
```
test_df_binarize.head()
```

Out[44]:

	ID	Name	DateTime	AnimalType	SexuponOutcome	AgeuponOutcome	Breed	Color
0	1	Summer	2015-10-12 12:15:00	Dog	Intact Female	43.0	Labrador Retriever	White
1	2	Cheyenne	2014-07-26 17:59:00	Dog	Spayed Female	204.0	Siberian Husky	Tan
2	3	Gus	2016-01-13 12:20:00	Cat	Neutered Male	52.0	Domestic Shorthair	Brown Tabby
3	4	Pongo	2013-12-28 18:12:00	Dog	Intact Male	17.0	Collie Smooth	Tricolor
4	5	Skooter	2015-09-24 17:59:00	Dog	Neutered Male	204.0	Miniature Poodle	White

5 rows × 270 columns

In [45]:
```
test_df_binarize.rename(columns = {'ID':'AnimalID'}, inplace = True)
```

In [46]:
```
# Print full list of columns from binarization to check work
print("Num of columns:", len(list(test_df_binarize.columns.values)),
      "\nand columns:", list(test_df_binarize.columns.values),end="")
```

```
Num of columns: 270
and columns: ['AnimalID', 'Name', 'DateTime', 'AnimalType', 'SexuponOutcome',
'AgeuponOutcome', 'Breed', 'Color', 'State', 'Gender', 'AnimalType_Binarize(D
og=1)', 'State(Fixed=1)', 'Gender(Male=1)', 'Abyssinian', 'Airedale Terrier',
'Akita', 'Alaskan Husky', 'Alaskan Malamute', 'American Bulldog', 'American E
skimo', 'American Foxhound', 'American Pit Bull Terrier', 'American Shorthai
r', 'American Staffordshire Terrier', 'American Wirehair', 'Anatol Shepherd',
'Angora', 'Australian Cattle Dog', 'Australian Kelpie', 'Australian Shepher
d', 'Australian Terrier', 'Balinese', 'Basenji', 'Basset Hound', 'Beagle', 'B
earded Collie', 'Beauceron', 'Belgian Malinois', 'Belgian Tervuren', 'Benga
l', 'Bernese Mountain Dog', 'Bichon Frise', 'Black Mouth Cur', 'Black/Tan Hou
nd', 'Bloodhound', 'Blue Lacy', 'Bluetick Hound', 'Boerboel', 'Bombay', 'Bord
er Collie', 'Border Terrier', 'Boston Terrier', 'Boxer', 'Boykin Span', 'Brit
ish Shorthair', 'Brittany', 'Bruss Griffon', 'Bull Terrier', 'Bull Terrier Mi
niature', 'Bulldog', 'Bullmastiff', 'Cairn Terrier', 'Canaan Dog', 'Cane Cors
o', 'Cardigan Welsh Corgi', 'Carolina Dog', 'Catahoula', 'Cavalier Span', 'Ch
artreux', 'Chesa Bay Retr', 'Chihuahua Longhair', 'Chihuahua Shorthair', 'Chi
nese Crested', 'Chinese Sharpei', 'Chow Chow', 'Cirneco', 'Cocker Spaniel',
'Collie Rough', 'Collie Smooth', 'Coton De Tulear', 'Cymric', 'Dachshund', 'D
achshund Longhair', 'Dachshund Stan', 'Dachshund Wirehair', 'Dalmatian', 'Dan
die Dinmont', 'Devon Rex', 'Doberman Pinsch', 'Dogo Argentino', 'Dogue De Bor
deaux', 'Domestic Longhair', 'Domestic Medium Hair', 'Domestic Shorthair', 'D
utch Shepherd', 'Eng Toy Spaniel', 'English Bulldog', 'English Cocker Spanie
l', 'English Coonhound', 'English Foxhound', 'English Pointer', 'English Spri
nger Spaniel', 'Feist', 'Field Spaniel', 'Finnish Spitz', 'Flat Coat Retrieve
r', 'French Bulldog', 'German Pinscher', 'German Shepherd', 'German Shorthair
Pointer', 'German Wirehaired Pointer', 'Glen Of Imaal', 'Golden Retriever',
'Gordon Setter', 'Great Dane', 'Great Pyrenees', 'Greater Swiss Mountain Do
g', 'Greyhound', 'Harrier', 'Havanese', 'Himalayan', 'Ibizan Hound', 'Irish S
etter', 'Irish Terrier', 'Irish Wolfhound', 'Italian Greyhound', 'Jack Russel
l Terrier', 'Japanese Chin', 'Keeshond', 'Kuvasz', 'Labrador Retriever', 'Leo
nberger', 'Lhasa Apso', 'Maine Coon', 'Maltese', 'Manchester Terrier', 'Man
x', 'Mastiff', 'Mexican Hairless', 'Miniature Pinscher', 'Miniature Poodle',
'Miniature Schnauzer', 'Munchkin Shorthair', 'Norfolk Terrier', 'Norwegian Fo
rest Cat', 'Norwich Terrier', 'Nova Scotia Duck Tolling Retriever', 'Old Engl
ish Bulldog', 'Old English Sheepdog', 'Oriental Sh', 'Otterhound', 'Papillo
n', 'Parson Russell Terrier', 'Patterdale Terr', 'Pbgv', 'Pekingese', 'Pembro
ke Welsh Corgi', 'Persian', 'Pharaoh Hound', 'Picardy Sheepdog', 'Pit Bull',
'Pixiebob Shorthair', 'Plott Hound', 'Podengo Pequeno', 'Pointer', 'Pomerania
n', 'Presa Canario', 'Pug', 'Queensland Heeler', 'Ragdoll', 'Rat Terrier', 'R
edbone Hound', 'Rex', 'Rhod Ridgeback', 'Rottweiler', 'Russian Blue', 'Saluk
i', 'Samoyed', 'Schipperke', 'Schnauzer Giant', 'Scottish Fold', 'Scottish Te
rrier', 'Shetland Sheepdog', 'Shiba Inu', 'Shih Tzu', 'Siamese', 'Siberian Hu
sky', 'Silky Terrier', 'Skye Terrier', 'Smooth Fox Terrier', 'Snowshoe', 'Sof
t Coated Wheaten Terrier', 'St. Bernard Rough Coat', 'St. Bernard Smooth Coa
t', 'Staffordshire', 'Standard Poodle', 'Standard Schnauzer', 'Swedish Vallhu
nd', 'Tan Hound', 'Tan Hound/Siberian Husky', 'Tibetan Spaniel', 'Tibetan Ter
rier', 'Tonkinese', 'Toy Fox Terrier', 'Toy Poodle', 'Treeing Cur', 'Treeing
Walker Coonhound', 'Turkish Angora', 'Vizsla', 'Weimaraner', 'Welsh Terrier',
'West Highland', 'Whippet', 'Wire Hair Fox Terrier', 'Wirehaired Pointing Gri
ffon', 'Yorkshire Terrier', 'Agouti', 'Apricot', 'Black', 'Black Brindle', 'B
lack Smoke', 'Black Tabby', 'Blue', 'Blue Cream', 'Blue Merle', 'Blue Point',
'Blue Smoke', 'Blue Tabby', 'Blue Tick', 'Blue Tiger', 'Brown', 'Brown Brindl
e', 'Brown Merle', 'Brown Tabby', 'Buff', 'Calico', 'Calico Point', 'Chocolat
e', 'Chocolate Point', 'Cream', 'Cream Tabby', 'Fawn', 'Flame Point', 'Gold',
'Gray', 'Gray Tabby', 'Gray Tiger', 'Lilac Point', 'Liver', 'Lynx Point', 'Or
ange', 'Orange Tabby', 'Pink', 'Red', 'Red Merle', 'Red Tick', 'Sable', 'Seal
```

```
Point', 'Silver', 'Silver Lynx Point', 'Silver Tabby', 'Tan', 'Torbie', 'Tort
ie', 'Tortie Point', 'Tricolor', 'White', 'Yellow', 'Yellow Brindle', 'Ageupo
nOutcome_norm']
```

In [47]:
```
# Examine the unique columns in train_df_binarize and test_df_binarize.
# We will eventually want all columns in training to also be accounted for in
 test.
x = set(train_df_binarize.columns).symmetric_difference(test_df_binarize.colum
ns)

# Parse out only the unique items in both lists within the training list and t
est list, respectively.
train_unique = [value for value in list(x)
                if value in list(train_df_binarize.columns.values)]
test_unique = [value for value in list(x)
               if value in list(test_df_binarize.columns.values)]

print ("Number of unique columns in the train dataset not present in the test
 dataset:", len(train_unique))
print ("Number of unique columns in the test dataset not present in the train
 dataset:", len(test_unique))
```

```
Number of unique columns in the train dataset not present in the test datase
t: 42
Number of unique columns in the test dataset not present in the train datase
t: 16
```

We will rectify the issue of the training and test datasets not having the same number of features by adding features initialized with zero to the test dataset. We will also remove features exclusive to the test dataset.

In [48]:
```
# Add all of the unique columns in the training dataset to the test dataset an
d initialize as 0.
for col in train_unique:
    test_df_binarize[col] = 0

print ("New number of columns in the test binarization dataset adding train on
ly columns:",
       len(list(test_df_binarize.columns.values)))
```

```
New number of columns in the test binarization dataset adding train only colu
mns: 312
```

In [49]:
```
# Remove all of the unique columns in the test dataset that are not also in th
e training dataset.
test_df_binarize = test_df_binarize.drop([col for col in test_unique], axis=1)

print ("New number of columns in the test binarization dataset removing test o
nly columns:",
       len(list(test_df_binarize.columns.values)))
```

```
New number of columns in the test binarization dataset removing test only col
umns: 296
```

In [50]:
```
# Export a CSV record of the dataset.
test_df_binarize.to_csv('data/test_df_binarization.csv')
```

Literature Review

Next, we looked at our literature. From our EDA, we realized that we had hundreds of features, even after reduction, many of which were the very sparse dummy variables for breed and color. We also realized that we had very unbalanced classes, which would make prediction difficult with some algorithms, given that our test dataset would likely not have the same balance as our training dataset. From our class readings and algorithm review, we thought random forests may be our best bet, so we began our literature investigation there.

Many of our papers discussed the relative merits of using logistic regression or random forest classifiers. One particular article, dealing with storm data, taught us about two performance scores that may work even better than receiver operating characteristic curves for highly unbalanced data: false alarm rate and threat score, (Ruiz, Anne, and Villa, Nathalie). Other papers discussed the benefits of the random forest procedure over a single decision tree classifier. Especially for applications in which there are a large number of features, but not all observations have all of the features, random forests win out over individual decision trees, (Breiman, Leo).

Other articles helped us learn more about decision trees and random forests holistically. For instance, we were able to understand more about the advantages and disadvantages between tree complexity and the goodness of fit, which would be helpful for our models as well, (Song, Yan-yan, and Lu, Ying). We also learned that, while the weight of feature importance varies within each tree, order of feature importance usually does not. Therefore, random forests can likely provide us with good outcomes, (Liaw, Andy and Wiener, Matthew).

With regards to the potential data imputation process, we read about the merits of oversampling and undersampling. For instance, we opted to oversample from our underrepresented outcome classes and undersample from our overrepresented outcome classes to try and balance the data for better predictions (Galar, M., A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera).

Setting up data for model building

In [51]:
```
# Reorder the test data columns to be in the same order as the train data colu
mns.

train_cols = train_df_binarize.columns.tolist()
test_df_binarize_ordered = test_df_binarize[train_cols]
```

In [52]:
```
# We need to separate the training data and labels and add them to a pandas da
taframe.
train_data_pd = train_df_binarize.iloc[: , 13:]
train_labels_pd = train_df_binarize['OutcomeType']

train_data_pd.head()
```

Out[52]:

	State(Fixed=1)	Gender(Male=1)	Abyssinian	Affenpinscher	Afghan Hound	Airedale Terrier	Akita	Alaskan Husky
0	1	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0

5 rows × 283 columns

Within our model building process, we will first turn our binarized train and test data frames into numpy arrays for efficiency. We will also remove our outcome variable from the training dataset and create a separate label array.

In [53]:
```
# We need a numpy array to run the models efficiently, so we convert the train
ing data here.
train_data = train_data_pd.values
train_data
```

Out[53]:
```
array([[1.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.05      ],
       [1.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.05      ],
       [1.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.19615385],
       ...,
       [1.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.2       ],
       [0.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.00384615],
       [0.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.05      ]])
```

In [54]:
```
# We need a numpy array to run the models efficiently, so we convert the train
ing labels here.
train_labels = train_labels_pd.values
train_labels
```

Out[54]:
```
array(['Return_to_owner', 'Euthanasia', 'Adoption', ..., 'Adoption',
       'Transfer', 'Transfer'], dtype=object)
```

In [55]:
```
# We need to separate the test data and add it to a pandas dataframe.
test_data_pd = test_df_binarize_ordered.iloc[: , 13:]

test_data_pd.head()
```

Out[55]:

	State(Fixed=1)	Gender(Male=1)	Abyssinian	Affenpinscher	Afghan Hound	Airedale Terrier	Akita	Alaskan Husky
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0

5 rows × 283 columns

In [56]:
```
# We need a numpy array to run the models efficiently, so we convert the test
 data here.
test_data = test_data_pd.values
test_data
```

Out[56]:
```
array([[0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.04134615],
       [1.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.19615385],
       [1.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.05      ],
       ...,
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.05      ],
       [1.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.3       ],
       [0.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.2       ]])
```

In [57]:
```
# Check train_data and train_labels shape.
print('train_data shape:', train_data.shape,
      '\ntrain_labels shape:', train_labels.shape,
      '\ntest_data shape:', test_data.shape
     )
```

```
train_data shape: (26729, 283)
train_labels shape: (26729,)
test_data shape: (11456, 283)
```

Model Creation

Baseline model - the "Dumb" classifier

If we were to fit a model that predicted that all outcomes were the majority class, we would get a classification report seen below. This model is not a good one, but this will be the baseline against which we test our performance and accuracy improvements. We will evaluate all of our models below using the f1-score. We decided that accuracy, which is simply the proportion of observations predicted correctly over total observations, is not the best measure. Accuracy assigns equal weight to false positives and false negatives. When the dataset is balanced, that can work very well. However, as we discovered earlier, our dataset is not balanced. Therefore, we would still have mediocre accuracy if all observations were predicted as the adoption class (which we will see below).

We will use the weighted f1-score to evaluate prediction performance. The weighted f1-score takes a proportional weight of the f1-score of all classes to calculate the weighted f1-score. The f1-score itself takes precision (ratio of the true positives to true and false positives) and recall (ratio of the true positives to true positives and false negatives) into account. Therefore, the f1-score is essentially taking both false positives and false negatives into account without needing to weight them equally. This type of scoring system to evaluate prediction performance works best when classes are not balanced.

In [58]:
```
# Set all predicted values to "Adoption".
pred_labels = train_labels.copy()
pred_labels.fill('Adoption')
print(classification_report(train_labels, pred_labels))
print('Accuracy = {:.2f}'.format(np.mean(train_labels == pred_labels)))
f1_dict = {}
f1_dict['Dumb'] = ['%0.2f' % metrics.f1_score(train_labels, pred_labels, avera
ge='micro'),
                   '%0.2f' % metrics.f1_score(train_labels, pred_labels, avera
ge='weighted'), '']
```

```
                precision    recall  f1-score   support

      Adoption       0.40      1.00      0.57     10769
          Died       0.00      0.00      0.00       197
     Euthanasia       0.00      0.00      0.00      1555
Return_to_owner       0.00      0.00      0.00      4786
      Transfer       0.00      0.00      0.00      9422

     micro avg       0.40      0.40      0.40     26729
     macro avg       0.08      0.20      0.11     26729
  weighted avg       0.16      0.40      0.23     26729

Accuracy = 0.40
```

If we had a model that predicted just adoption, we would have an accuracy of 0.40, and a weighted f1-score of 0.23.

Finding and Choosing the Best Algorithm

In order to decide which algorithm to use, we need to test several approaches. Our first step is to create a function, using the `StratifiedKFold` library within sckit-learn, to split our training dataset into training and test. This function randomly splits the training and test data (for our purposes, development data) a specified number of times. Therefore, we do not need to create a separate development dataset. We will be optimizing the weighted f1-score utilizing the StratifiedKFold function.

The StratifiedKFold trains the classifiers on about 90% of the training data and then predicts on 10% of the training data. We decided to pick the StratifiedKFold function over the KFold function, as the StratifiedKFold function picks a proportional distribution, while the KFold function simply selects from the distribution. As our classes were so unbalanced, we believed that the StratifiedKFold function would give us a more representative sampling.

We decided to first start with five separate predictions to check if the accuracy changed each time with logistic regression, decision trees, random forests. Since we are working with a sparse data set, we researched additional classifiers that might perform well with that constraint. The only recommendation we found with which we were not familar was XGboost, which builds upon the decision tree approach with a gradient-boosted decision trees method. Gradient boosting constructs new decision tree models that predict the residuals of prior models and then adds them together to make final predictions. It uses a gradient descent algorithm to minimize loss when adding in new models (Chen, Tianqi and Guestrin, Carlos). While this algorithm is fast compared to other gradient boosted tree classifiers, it is extraordinarily slow compared to the other methods we are using. However, it usually does well with large-scale datasets.

We will print classification reports at each stage to understand more about each algorithm, as well as other performance measures. Along with the StratifiedKFold function, we also set up functions to ensure that we are printing classification reports and confusion matrices where necessary. Lastly, we will implement a function to check the variance between each of the training and test datasets.

Since our dataset contains labels, we are steering away from unsupervised clustering approaches, such as K Means and Gaussian Mixture models, and focusing on supervised approaches.

In [59]:
```
def print_matrix(cm, title):
    '''This function takes in a confusion matrix, and plots it.'''
    n = cm.shape[0]
    fig = plt.figure()
    ax = fig.add_subplot(111)
    cax = ax.matshow(cm, cmap = 'Blues')
    plt.title('Confusion Matrix, {}'.format(title))
    fig.colorbar(cax)
    plt.xticks(np.arange(0, n, step=1))
    plt.yticks(np.arange(0, n, step=1))
    for x in range(n):
        for y in range(n):
            plt.text(x,y,str(cm[y][x]), horizontalalignment='center')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    return fig
```

In [60]:

```
def stratfit(clf, train_data, train_labels, n_splits=10):
    '''This function fits train_data and train_labels to any classifier using
 StratifiedKFold.
    It prints a classification report, a confusion matrix.'''

    # Use StratifiedKFold here so that we can preserve the relative percentage
of classes in the folds.
    kf = StratifiedKFold(n_splits=n_splits, shuffle=True)
    kf.get_n_splits(train_data)

    # For the stratifiedkfolds, fit the classifier,
    # predict values and generate a classifier report for the training folds,
    # then predict values and generate a classifier report for the test folds.
    for index, [train_index, test_index] in enumerate(kf.split(train_data, tra
in_labels)):
        print("\nTRAIN:", len(train_index), "and TEST:", len(test_index))

        # Fit the model to the training folds.
        clf_out = clf.fit(train_data[train_index], train_labels[train_index])

        # Predict values for the training folds.
        pred_labels_train = clf.predict(train_data[train_index])

        # Print the classification report.
        print('\nFor train data:\n',
              classification_report(train_labels[train_index], pred_labels_tra
in))

        # Predict values for the test folds.
        pred_labels = clf.predict(train_data[test_index])

        # Print classification report.
        print('For test data:\n',
              classification_report(train_labels[test_index], pred_labels))

        # Print the confusion matrix.
        print_matrix(confusion_matrix(train_labels[test_index],
                                      pred_labels), 'Split {}'.format(index +
1))

        # Add weighted accuracy and f1 scores to temporary lists.
        accuracy_temp_list.append(metrics.f1_score(train_labels[test_index],
                                                   pred_labels, average='micr
o'))
        f1_temp_list.append(metrics.f1_score(train_labels[test_index],
                                             pred_labels, average='weighted'))
```

In [61]:
```
def check_variance(clf, X_train, y_train, n_splits=100):
    '''This function takes in a classifier (clf), data (X_train), labels (y_train),
    repeatedly fits the classifier to a stratified cross validation set,
    and returns the weighted f1_scores. Will use this function throughout.
    '''
    # Create shufflesplit that will split into n_splits folds, and put 20 % into the test
    cv = ShuffleSplit(n_splits=n_splits, test_size = 0.2)
    # Caclulate f1 score for each fit, and store it in cv_scores
    cv_scores = cross_val_score(clf, X_train, y_train, cv=cv, scoring='f1_weighted')

    return cv_scores
```

In [62]:
```
# Initiate dictionary to hold cv_scores for each kind of classifier.
cv_scores_dict = {}
```

In [63]:
```
# Initiate lists to hold accuracy and f1 scores while StratifiedKFold is calculating.
f1_temp_list = []
accuracy_temp_list = []
```

We start with logistic regression as the first attempted classifier, as we are familiar with it.

In [64]:
```
clf_LR = LogisticRegression()
stratfit(clf_LR, train_data, train_labels, n_splits=5)
cv_scores_dict['1. Logistic Regression\nwith Raw Data'] = check_variance(clf_L
R
                                                                          , tra
in_data
                                                                          , tra
in_labels
                                                                          , n_s
plits = 5
                                                                         )
f1_dict['Logistic Regression with Raw Data'] = ['%.2f' % np.mean(accuracy_temp
_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 21381 and TEST: 5348

For train data:

	precision	recall	f1-score	support
Adoption	0.62	0.89	0.73	8615
Died	0.00	0.00	0.00	157
Euthanasia	0.51	0.03	0.05	1244
Return_to_owner	0.47	0.29	0.36	3828
Transfer	0.73	0.62	0.67	7537
micro avg	0.63	0.63	0.63	21381
macro avg	0.47	0.37	0.36	21381
weighted avg	0.62	0.63	0.60	21381

For test data:

	precision	recall	f1-score	support
Adoption	0.61	0.89	0.72	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.39	0.02	0.04	311
Return_to_owner	0.47	0.29	0.36	958
Transfer	0.72	0.61	0.66	1885
micro avg	0.63	0.63	0.63	5348
macro avg	0.44	0.36	0.36	5348
weighted avg	0.60	0.63	0.59	5348

TRAIN: 21382 and TEST: 5347

For train data:

	precision	recall	f1-score	support
Adoption	0.61	0.89	0.73	8615
Died	0.00	0.00	0.00	157
Euthanasia	0.49	0.02	0.04	1244
Return_to_owner	0.47	0.29	0.36	3829
Transfer	0.72	0.62	0.67	7537
micro avg	0.63	0.63	0.63	21382
macro avg	0.46	0.36	0.36	21382
weighted avg	0.62	0.63	0.60	21382

For test data:

	precision	recall	f1-score	support
Adoption	0.61	0.89	0.73	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.67	0.02	0.04	311
Return_to_owner	0.47	0.28	0.35	957
Transfer	0.72	0.62	0.67	1885
micro avg	0.63	0.63	0.63	5347
macro avg	0.49	0.36	0.36	5347
weighted avg	0.62	0.63	0.59	5347

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.62	0.89	0.73	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.51	0.02	0.05	1244
Return_to_owner	0.48	0.30	0.37	3829
Transfer	0.72	0.62	0.67	7538
micro avg	0.63	0.63	0.63	21384
macro avg	0.47	0.37	0.36	21384
weighted avg	0.62	0.63	0.60	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.61	0.88	0.72	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.38	0.03	0.05	311
Return_to_owner	0.46	0.29	0.36	957
Transfer	0.74	0.63	0.68	1884
micro avg	0.63	0.63	0.63	5345
macro avg	0.44	0.37	0.36	5345
weighted avg	0.61	0.63	0.60	5345

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.61	0.89	0.73	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.41	0.02	0.04	1244
Return_to_owner	0.48	0.30	0.37	3829
Transfer	0.73	0.62	0.67	7538
micro avg	0.63	0.63	0.63	21384
macro avg	0.45	0.37	0.36	21384
weighted avg	0.61	0.63	0.60	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.62	0.89	0.73	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.53	0.03	0.05	311
Return_to_owner	0.45	0.29	0.35	957
Transfer	0.73	0.63	0.67	1884
micro avg	0.63	0.63	0.63	5345
macro avg	0.47	0.37	0.36	5345
weighted avg	0.62	0.63	0.60	5345

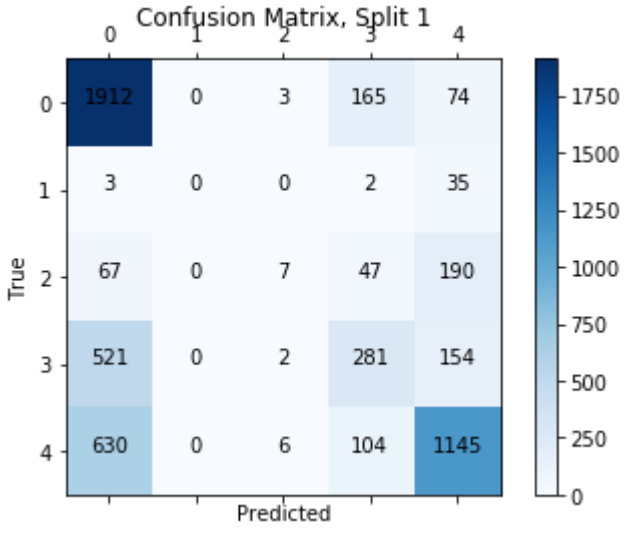
TRAIN: 21385 and TEST: 5344

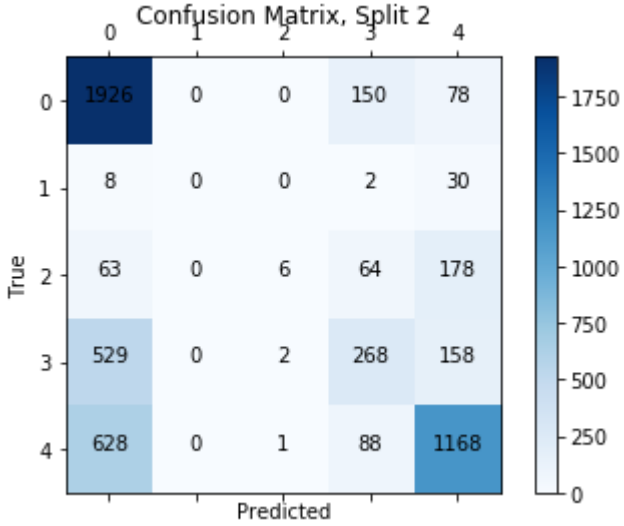
For train data:

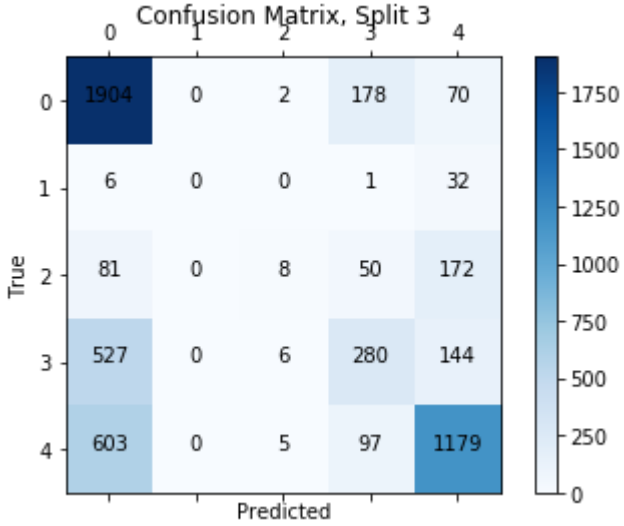
	precision	recall	f1-score	support
Adoption	0.61	0.89	0.73	8616
Died	0.00	0.00	0.00	158
Euthanasia	0.52	0.02	0.05	1244
Return_to_owner	0.48	0.30	0.37	3829
Transfer	0.73	0.62	0.67	7538
micro avg	0.63	0.63	0.63	21385
macro avg	0.47	0.37	0.36	21385
weighted avg	0.62	0.63	0.60	21385

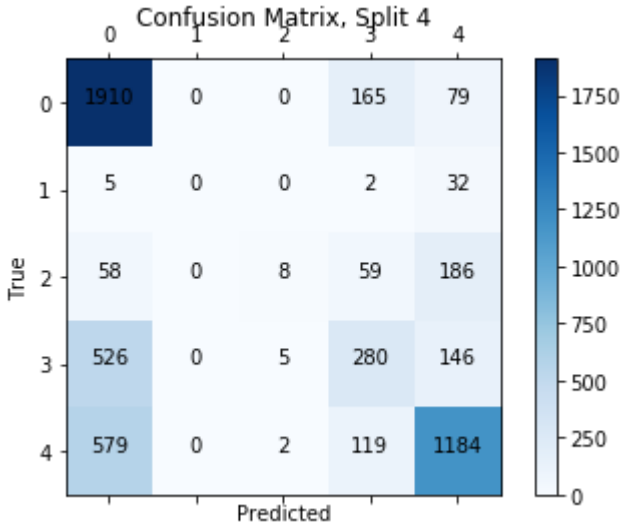
For test data:

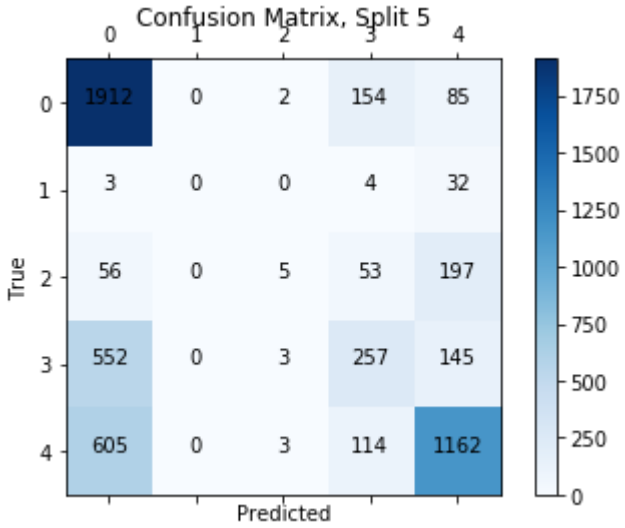
	precision	recall	f1-score	support
Adoption	0.61	0.89	0.72	2153
Died	0.00	0.00	0.00	39
Euthanasia	0.38	0.02	0.03	311
Return_to_owner	0.44	0.27	0.33	957
Transfer	0.72	0.62	0.66	1884
micro avg	0.62	0.62	0.62	5344
macro avg	0.43	0.36	0.35	5344
weighted avg	0.60	0.62	0.59	5344




Confusion Matrix, Split 2
0 1 2 3 4
0 1926 0 0 150 78
1 8 0 0 2 30
2 63 0 6 64 178
3 529 0 2 268 158
4 628 0 1 88 1168
True
Predicted
1750
1500
1250
1000
750
500
250
0


Confusion Matrix, Split 3
0 1 2 3 4
0 1904 0 2 178 70
1 6 0 0 1 32
2 81 0 8 50 172
3 527 0 6 280 144
4 603 0 5 97 1179
True
Predicted
1750
1500
1250
1000
750
500
250
0


Confusion Matrix, Split 4
0 1 2 3 4
0 1910 0 0 165 79
1 5 0 0 2 32
2 58 0 8 59 186
3 526 0 5 280 146
4 579 0 2 119 1184
True
Predicted
1750
1500
1250
1000
750
500
250
0



Our weighted f1-score is ~0.59 for the logistic regression classifier for test data. While this is an improvement over the base case, we think we can find a classifier that is better.

Next, let's evaluate a decision tree classifier.

In [65]:
```
clf_DT = DecisionTreeClassifier()
stratfit(clf_DT, train_data, train_labels, n_splits = 5)
cv_scores_dict['2. Decision Tree\nwith Raw Data'] = check_variance(clf_DT
                                                                  , train_dat
a
                                                                  , train_lab
els
                                                                  , n_splits
= 5
                                                                 )
f1_dict['Decision Tree with Raw Data'] = ['%.2f' % np.mean(accuracy_temp_list
), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 21381 and TEST: 5348

For train data:

	precision	recall	f1-score	support
Adoption	0.75	0.93	0.83	8615
Died	0.78	0.31	0.45	157
Euthanasia	0.78	0.50	0.61	1244
Return_to_owner	0.80	0.65	0.72	3828
Transfer	0.85	0.77	0.81	7537
micro avg	0.79	0.79	0.79	21381
macro avg	0.79	0.63	0.68	21381
weighted avg	0.80	0.79	0.79	21381

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.77	0.69	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.24	0.15	0.18	311
Return_to_owner	0.39	0.32	0.35	958
Transfer	0.68	0.62	0.65	1885
micro avg	0.59	0.59	0.59	5348
macro avg	0.39	0.37	0.38	5348
weighted avg	0.58	0.59	0.58	5348

TRAIN: 21382 and TEST: 5347

For train data:

	precision	recall	f1-score	support
Adoption	0.76	0.92	0.83	8615
Died	0.81	0.27	0.41	157
Euthanasia	0.80	0.49	0.61	1244
Return_to_owner	0.79	0.67	0.72	3829
Transfer	0.85	0.77	0.81	7537
micro avg	0.79	0.79	0.79	21382
macro avg	0.80	0.62	0.68	21382
weighted avg	0.80	0.79	0.79	21382

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.78	0.70	2154
Died	0.07	0.03	0.04	40
Euthanasia	0.20	0.11	0.14	311
Return_to_owner	0.39	0.35	0.37	957
Transfer	0.70	0.61	0.65	1885
micro avg	0.60	0.60	0.60	5347
macro avg	0.40	0.38	0.38	5347
weighted avg	0.59	0.60	0.59	5347

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.76	0.93	0.83	8615
Died	0.78	0.30	0.43	158
Euthanasia	0.78	0.51	0.62	1244
Return_to_owner	0.80	0.64	0.71	3829
Transfer	0.85	0.77	0.81	7538
micro avg	0.79	0.79	0.79	21384
macro avg	0.79	0.63	0.68	21384
weighted avg	0.80	0.79	0.79	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.62	0.77	0.69	2154
Died	0.09	0.03	0.04	39
Euthanasia	0.21	0.14	0.17	311
Return_to_owner	0.39	0.32	0.35	957
Transfer	0.68	0.61	0.64	1884
micro avg	0.59	0.59	0.59	5345
macro avg	0.40	0.37	0.38	5345
weighted avg	0.57	0.59	0.58	5345

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.76	0.92	0.83	8615
Died	0.71	0.30	0.42	158
Euthanasia	0.80	0.50	0.61	1244
Return_to_owner	0.78	0.66	0.72	3829
Transfer	0.85	0.78	0.81	7538
micro avg	0.79	0.79	0.79	21384
macro avg	0.78	0.63	0.68	21384
weighted avg	0.80	0.79	0.79	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.74	0.69	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.20	0.11	0.14	311
Return_to_owner	0.40	0.35	0.37	957
Transfer	0.66	0.63	0.64	1884
micro avg	0.59	0.59	0.59	5345
macro avg	0.38	0.37	0.37	5345
weighted avg	0.57	0.59	0.58	5345

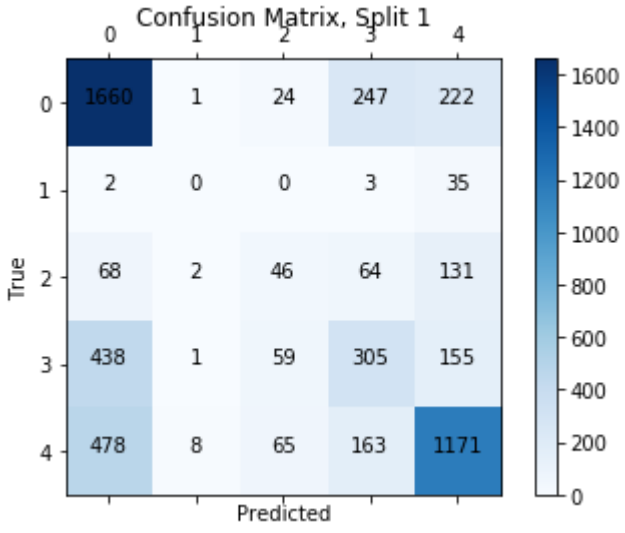
TRAIN: 21385 and TEST: 5344

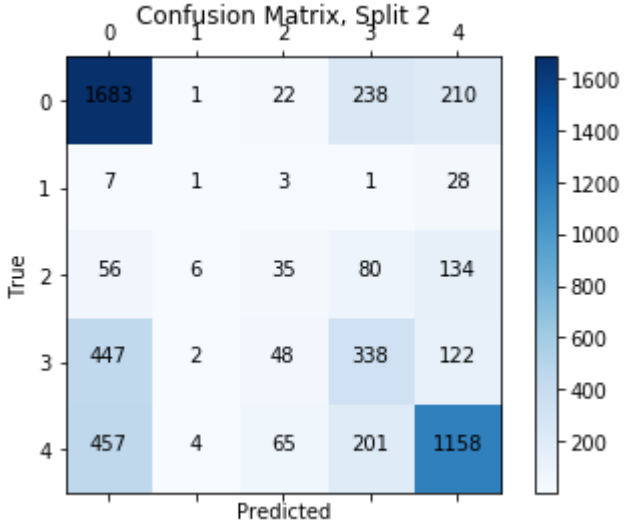
For train data:

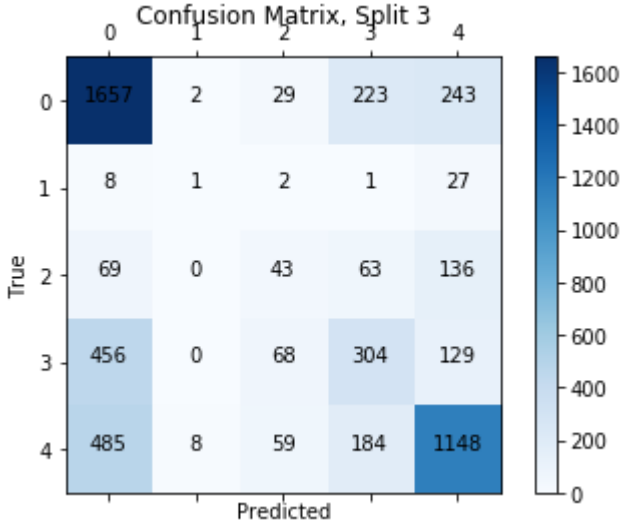
	precision	recall	f1-score	support
Adoption	0.75	0.93	0.83	8616
Died	0.83	0.28	0.42	158
Euthanasia	0.78	0.50	0.61	1244
Return_to_owner	0.81	0.65	0.72	3829
Transfer	0.85	0.77	0.81	7538
micro avg	0.79	0.79	0.79	21385
macro avg	0.81	0.63	0.68	21385
weighted avg	0.80	0.79	0.79	21385

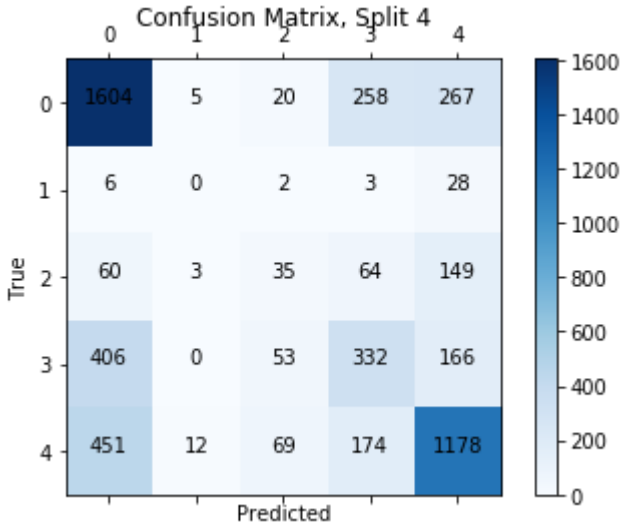
For test data:

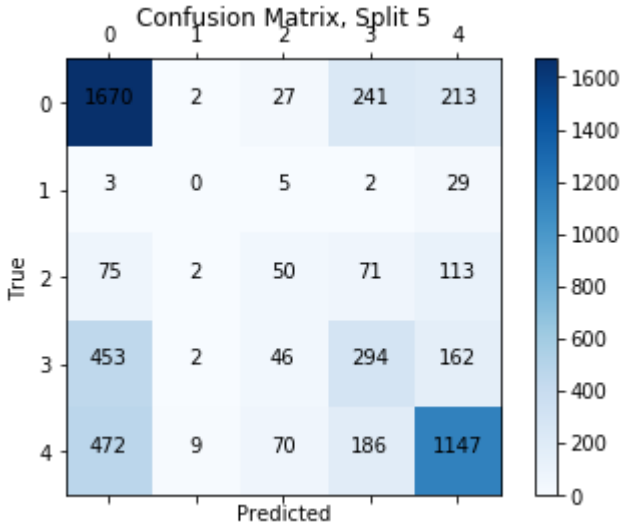
	precision	recall	f1-score	support
Adoption	0.62	0.78	0.69	2153
Died	0.00	0.00	0.00	39
Euthanasia	0.25	0.16	0.20	311
Return_to_owner	0.37	0.31	0.34	957
Transfer	0.69	0.61	0.65	1884
micro avg	0.59	0.59	0.59	5344
macro avg	0.39	0.37	0.37	5344
weighted avg	0.58	0.59	0.58	5344




Confusion Matrix, Split 2
0 1 2 3 4
0 1683 1 22 238 210
1 7 1 3 1 28
2 56 6 35 80 134
3 447 2 48 338 122
4 457 4 65 201 1158
True
Predicted
1600 1400 1200 1000 800 600 400 200


Confusion Matrix, Split 3
0 1 2 3 4
0 1657 2 29 223 243
1 8 1 2 1 27
2 69 0 43 63 136
3 456 0 68 304 129
4 485 8 59 184 1148
True
Predicted
1600 1400 1200 1000 800 600 400 200 0


Confusion Matrix, Split 4
0 1 2 3 4
0 1604 5 20 258 267
1 6 0 2 3 28
2 60 3 35 64 149
3 406 0 53 332 166
4 451 12 69 174 1178
True
Predicted
1600 1400 1200 1000 800 600 400 200 0



Our weighted f1-score on the test folds is ~0.58 for the decision tree classifier. This is slightly worse than the logistic regression classifier.

We also note that the f1-score for the test data (folds that were reserved for testing) is much lower than the training data (folds that were used for training), so we can say we have definitely poorly generalized the data for the decision tree classifier. The generalization problem points to the fact that the algorithm is not good at learning unseen data, as the prediction for the training data was far better than the prediction for the test data. We could counter the poor generalization problem by limiting the max_depth of the decision tree and setting a min_samples_split to be higher than the default of 2.

Next, we try the random forest classifier. We expect this to perform better than the decision tree out of the box, as this classifier takes the average of multiple decision trees, but we also think we will be able to tune it in order to get great performance in the end.

In [66]:
```
clf_RF = RandomForestClassifier()
stratfit(clf_RF, train_data, train_labels, n_splits=5)
cv_scores_dict['3. Random Forest\nwith Raw Data'] = check_variance(clf_RF
                                                                  , train_dat
a
                                                                  , train_lab
els
                                                                  , n_splits
= 5
                                                                 )
f1_dict['Random Forest with Raw Data'] = ['%.2f' % np.mean(accuracy_temp_list
), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 21381 and TEST: 5348

For train data:

	precision	recall	f1-score	support
Adoption	0.77	0.89	0.83	8615
Died	0.82	0.25	0.39	157
Euthanasia	0.78	0.47	0.59	1244
Return_to_owner	0.75	0.65	0.70	3828
Transfer	0.82	0.80	0.81	7537
micro avg	0.78	0.78	0.78	21381
macro avg	0.79	0.61	0.66	21381
weighted avg	0.79	0.78	0.78	21381

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.75	0.69	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.22	0.12	0.15	311
Return_to_owner	0.43	0.36	0.39	958
Transfer	0.66	0.64	0.65	1885
micro avg	0.60	0.60	0.60	5348
macro avg	0.39	0.37	0.38	5348
weighted avg	0.58	0.60	0.59	5348

TRAIN: 21382 and TEST: 5347

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.88	0.82	8615
Died	0.83	0.25	0.39	157
Euthanasia	0.79	0.46	0.58	1244
Return_to_owner	0.75	0.68	0.71	3829
Transfer	0.81	0.80	0.81	7537
micro avg	0.79	0.79	0.79	21382
macro avg	0.79	0.61	0.66	21382
weighted avg	0.79	0.79	0.78	21382

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.76	0.69	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.29	0.12	0.17	311
Return_to_owner	0.38	0.34	0.36	957
Transfer	0.66	0.63	0.65	1885
micro avg	0.60	0.60	0.60	5347
macro avg	0.39	0.37	0.37	5347
weighted avg	0.58	0.60	0.58	5347

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.88	0.83	8615
Died	0.79	0.24	0.37	158
Euthanasia	0.80	0.46	0.58	1244
Return_to_owner	0.75	0.67	0.71	3829
Transfer	0.81	0.80	0.81	7538
micro avg	0.79	0.79	0.79	21384
macro avg	0.79	0.61	0.66	21384
weighted avg	0.79	0.79	0.78	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.75	0.69	2154
Died	0.09	0.03	0.04	39
Euthanasia	0.21	0.10	0.14	311
Return_to_owner	0.39	0.33	0.36	957
Transfer	0.67	0.65	0.66	1884
micro avg	0.60	0.60	0.60	5345
macro avg	0.40	0.37	0.38	5345
weighted avg	0.58	0.60	0.58	5345

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.76	0.89	0.82	8615
Died	0.87	0.26	0.40	158
Euthanasia	0.80	0.46	0.58	1244
Return_to_owner	0.77	0.66	0.71	3829
Transfer	0.82	0.79	0.80	7538
micro avg	0.78	0.78	0.78	21384
macro avg	0.80	0.61	0.66	21384
weighted avg	0.79	0.78	0.78	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.76	0.69	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.26	0.11	0.16	311
Return_to_owner	0.41	0.32	0.36	957
Transfer	0.66	0.66	0.66	1884
micro avg	0.60	0.60	0.60	5345
macro avg	0.39	0.37	0.37	5345
weighted avg	0.58	0.60	0.59	5345

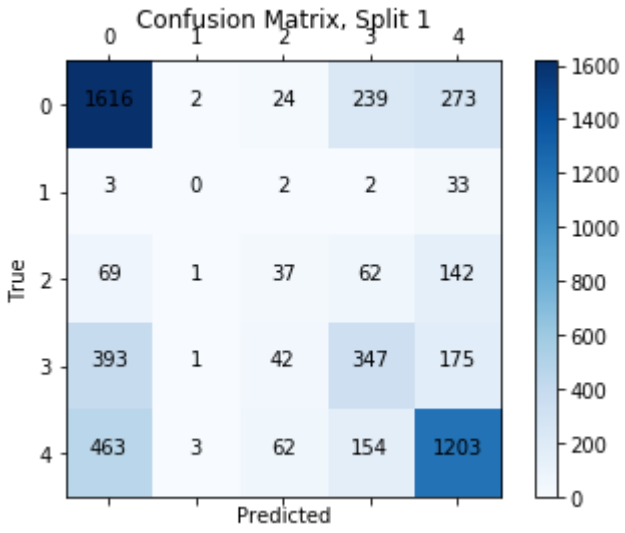
TRAIN: 21385 and TEST: 5344

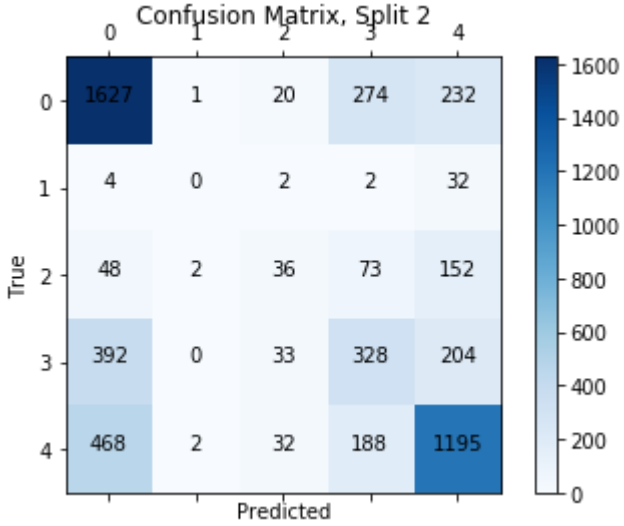
For train data:

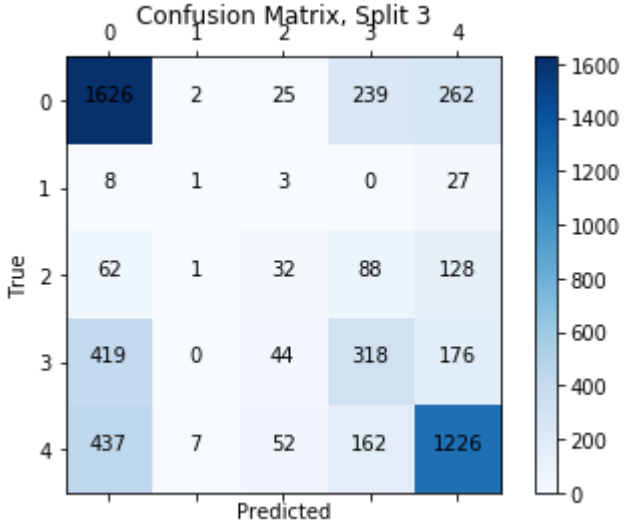
	precision	recall	f1-score	support
Adoption	0.77	0.88	0.82	8616
Died	0.70	0.25	0.37	158
Euthanasia	0.80	0.44	0.57	1244
Return_to_owner	0.76	0.65	0.70	3829
Transfer	0.81	0.80	0.80	7538
micro avg	0.78	0.78	0.78	21385
macro avg	0.77	0.61	0.65	21385
weighted avg	0.78	0.78	0.78	21385

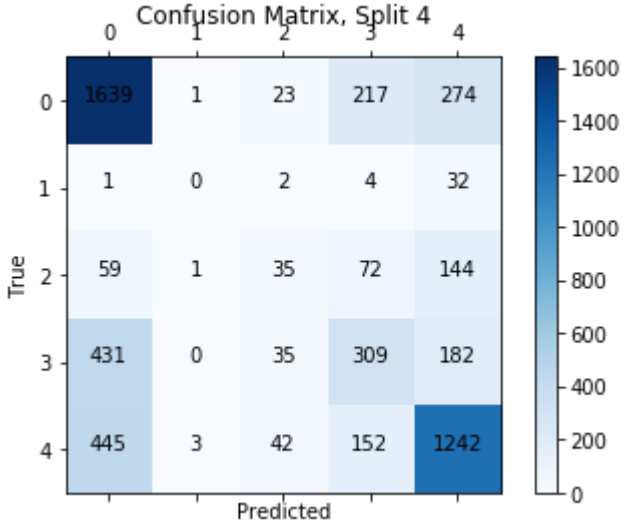
For test data:

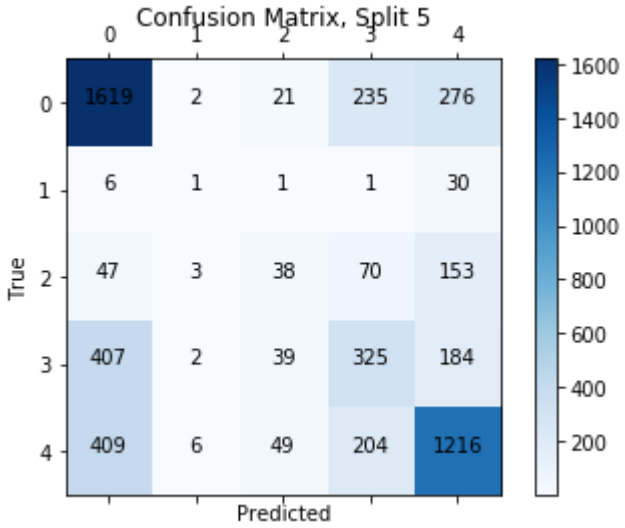
	precision	recall	f1-score	support
Adoption	0.65	0.75	0.70	2153
Died	0.07	0.03	0.04	39
Euthanasia	0.26	0.12	0.17	311
Return_to_owner	0.39	0.34	0.36	957
Transfer	0.65	0.65	0.65	1884
micro avg	0.60	0.60	0.60	5344
macro avg	0.40	0.38	0.38	5344
weighted avg	0.58	0.60	0.59	5344




Confusion Matrix, Split 2
0 1 2 3 4
0 1627 1 20 274 232
1 4 0 2 2 32
2 48 2 36 73 152
3 392 0 33 328 204
4 468 2 32 188 1195
True
Predicted
1600 1400 1200 1000 800 600 400 200 0


Confusion Matrix, Split 3
0 1 2 3 4
0 1626 2 25 239 262
1 8 1 3 0 27
2 62 1 32 88 128
3 419 0 44 318 176
4 437 7 52 162 1226
True
Predicted
1600 1400 1200 1000 800 600 400 200 0


Confusion Matrix, Split 4
0 1 2 3 4
0 1639 1 23 217 274
1 1 0 2 4 32
2 59 1 35 72 144
3 431 0 35 309 182
4 445 3 42 152 1242
True
Predicted
1600 1400 1200 1000 800 600 400 200 0



Our weighted f1-score is ~0.58 for the random forest classifier. This is in line with the decision tree classifier and slightly worse than the logistic regression classifier.

As we see with decision trees, the f1-scores for the test data (folds that were reserved for testing) is much lower than the train data (folds that were used for training), so we can say we have poorly generalized the data for the random forest classifier. We can counter the poor generalization problem by limiting the max_depth of the random forest and setting a min_samples_split to be higher than the default of 2. We will do this in the `GridSearch` during the optimization step.

Next, we will try out the XGBoost classifier.

In [67]:
```
clf_XGB = xgb.XGBClassifier()
stratfit(clf_XGB, train_data, train_labels, n_splits=5)
cv_scores_dict['4. XGBoost\nwith Raw Data'] = check_variance(clf_XGB
                                                             , train_dat
a
                                                             , train_lab
els
                                                             , n_splits
= 5
                                                            )
f1_dict['XGBoost with Raw Data'] = ['%.2f' % np.mean(accuracy_temp_list), '%.2
f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 21381 and TEST: 5348

For train data:

	precision	recall	f1-score	support
Adoption	0.63	0.87	0.73	8615
Died	0.00	0.00	0.00	157
Euthanasia	0.60	0.07	0.13	1244
Return_to_owner	0.47	0.39	0.43	3828
Transfer	0.76	0.61	0.68	7537
micro avg	0.64	0.64	0.64	21381
macro avg	0.49	0.39	0.39	21381
weighted avg	0.64	0.64	0.62	21381

For test data:

	precision	recall	f1-score	support
Adoption	0.61	0.86	0.72	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.60	0.08	0.15	311
Return_to_owner	0.43	0.35	0.39	958
Transfer	0.75	0.60	0.67	1885
micro avg	0.63	0.63	0.63	5348
macro avg	0.48	0.38	0.38	5348
weighted avg	0.62	0.63	0.60	5348

TRAIN: 21382 and TEST: 5347

For train data:

	precision	recall	f1-score	support
Adoption	0.63	0.87	0.73	8615
Died	0.00	0.00	0.00	157
Euthanasia	0.60	0.08	0.14	1244
Return_to_owner	0.46	0.38	0.42	3829
Transfer	0.75	0.61	0.67	7537
micro avg	0.64	0.64	0.64	21382
macro avg	0.49	0.39	0.39	21382
weighted avg	0.63	0.64	0.61	21382

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.87	0.73	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.47	0.05	0.09	311
Return_to_owner	0.45	0.37	0.41	957
Transfer	0.75	0.61	0.68	1885
micro avg	0.64	0.64	0.64	5347
macro avg	0.46	0.38	0.38	5347
weighted avg	0.63	0.64	0.61	5347

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.63	0.87	0.73	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.61	0.08	0.14	1244
Return_to_owner	0.47	0.38	0.42	3829
Transfer	0.75	0.61	0.67	7538
micro avg	0.64	0.64	0.64	21384
macro avg	0.49	0.39	0.39	21384
weighted avg	0.64	0.64	0.61	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.62	0.87	0.72	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.44	0.05	0.09	311
Return_to_owner	0.46	0.37	0.41	957
Transfer	0.75	0.61	0.67	1884
micro avg	0.63	0.63	0.63	5345
macro avg	0.45	0.38	0.38	5345
weighted avg	0.62	0.63	0.61	5345

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.62	0.88	0.73	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.49	0.08	0.13	1244
Return_to_owner	0.47	0.37	0.41	3829
Transfer	0.76	0.61	0.68	7538
micro avg	0.64	0.64	0.64	21384
macro avg	0.47	0.39	0.39	21384
weighted avg	0.63	0.64	0.61	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.62	0.88	0.73	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.38	0.06	0.10	311
Return_to_owner	0.47	0.36	0.41	957
Transfer	0.75	0.60	0.67	1884
micro avg	0.63	0.63	0.63	5345
macro avg	0.44	0.38	0.38	5345
weighted avg	0.62	0.63	0.61	5345

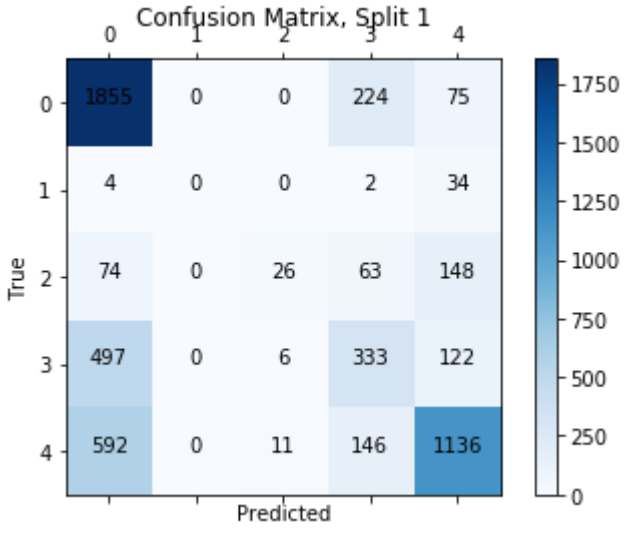
TRAIN: 21385 and TEST: 5344

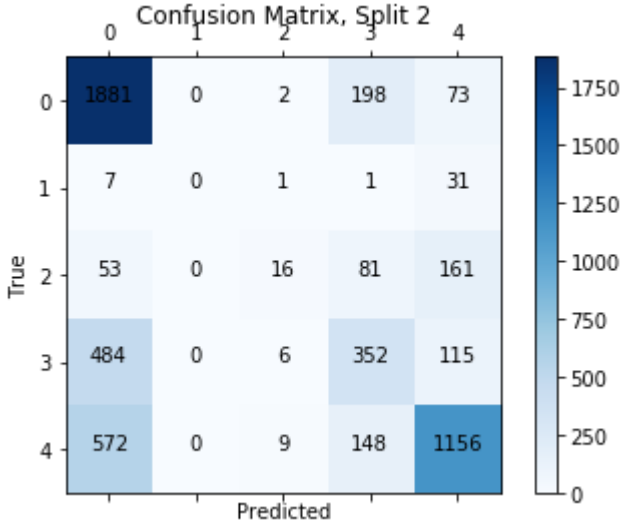
For train data:

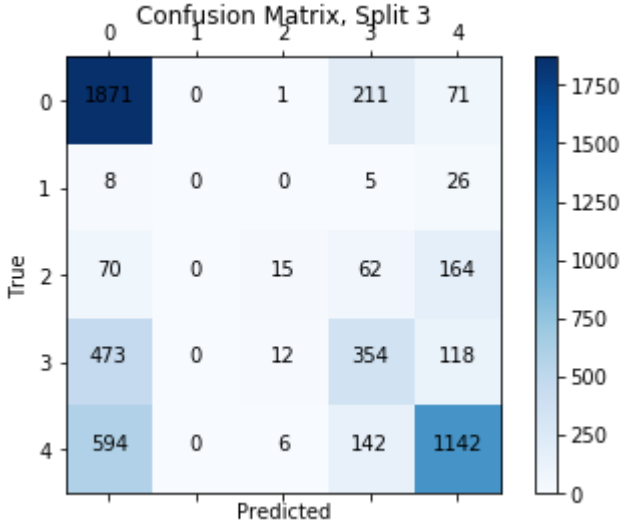
	precision	recall	f1-score	support
Adoption	0.63	0.87	0.73	8616
Died	0.00	0.00	0.00	158
Euthanasia	0.64	0.06	0.11	1244
Return_to_owner	0.47	0.38	0.42	3829
Transfer	0.75	0.61	0.67	7538
micro avg	0.64	0.64	0.64	21385
macro avg	0.50	0.39	0.39	21385
weighted avg	0.64	0.64	0.61	21385

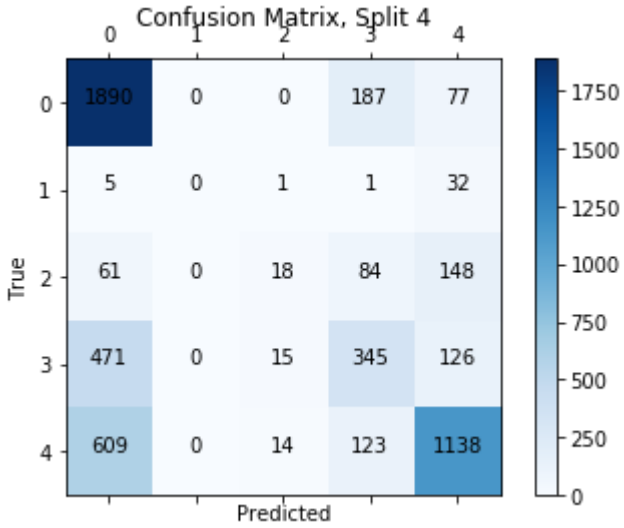
For test data:

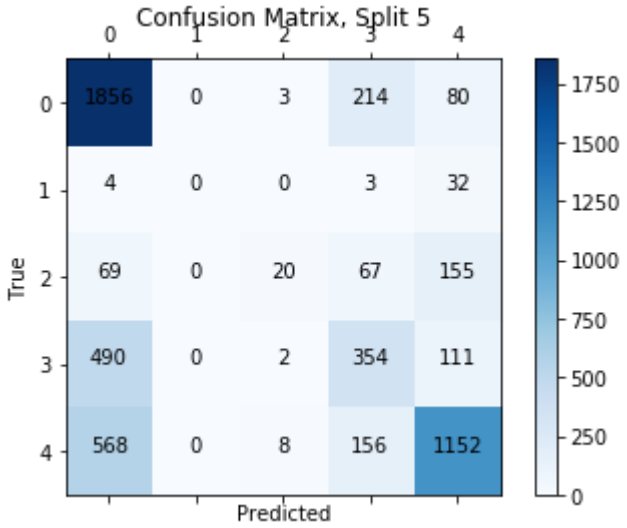
	precision	recall	f1-score	support
Adoption	0.62	0.86	0.72	2153
Died	0.00	0.00	0.00	39
Euthanasia	0.61	0.06	0.12	311
Return_to_owner	0.45	0.37	0.40	957
Transfer	0.75	0.61	0.67	1884
micro avg	0.63	0.63	0.63	5344
macro avg	0.49	0.38	0.38	5344
weighted avg	0.63	0.63	0.61	5344




Confusion Matrix, Split 2
0 1 2 3 4
0 1881 0 2 198 73
1 7 0 1 1 31
2 53 0 16 81 161
3 484 0 6 352 115
4 572 0 9 148 1156
True
Predicted
1750
1500
1250
1000
750
500
250
0


Confusion Matrix, Split 3
0 1 2 3 4
0 1871 0 1 211 71
1 8 0 0 5 26
2 70 0 15 62 164
3 473 0 12 354 118
4 594 0 6 142 1142
True
Predicted
1750
1500
1250
1000
750
500
250
0


Confusion Matrix, Split 4
0 1 2 3 4
0 1890 0 0 187 77
1 5 0 1 1 32
2 61 0 18 84 148
3 471 0 15 345 126
4 609 0 14 123 1138
True
Predicted
1750
1500
1250
1000
750
500
250
0



Though XGBoost runs very slowly, it returned a weighted f1-score is ~0.61, which is slightly better than all other outcomes thus far. This result tells us that XGBoost is worth further study.

We will examine logistic regression, random forests, and XGBoost as we refine our parameters and features. We will not continue with individual decision trees.

Throughout all of our analysis, the weighted f1-score was very consistent through each StratifiedKFold split, but the generalization issue highlighted our aforementioned data imbalance concerns with some of our outcome classes. We believe that if we work to over or under sample, we might be able to help our algorithm generalize to unseen data. The variance between each f1-score within each model was not large, as all f1-scores were close to each other within each classifier, so we are not incredibly worried about overfitting. However, we will perform further cross-validation between the test folds to evaluate our specific amount of overfitting.

Correcting for Class Imbalance

Since our data is so imbalanced, and we observe very few deaths and euthanasia outcomes, our models predict these outcomes quite poorly. We attempt to deal with this via oversampling then undersampling.

We used the RandomOverSampler() function to oversample from classes with low quantities of outcomes.

In [68]:
```
# Oversample to balance outcome classes.
ros = RandomOverSampler(random_state=0)
train_data_os, train_labels_os = ros.fit_resample(train_data, train_labels)

# Count to double check how many animals are in each class after we employ the
RandomOverSampler.
print(sorted(Counter(train_labels_os).items()))
```

```
[('Adoption', 10769), ('Died', 10769), ('Euthanasia', 10769), ('Return_to_own
er', 10769), ('Transfer', 10769)]
```

The result of this function is a near even amount of animals in each outcome class. Now that we have a balanced dataset, let's re-try the classifiers that we wanted to explore further: Logistic Regression, Random Forest, and XGBoost.

In [69]:

```
clf_LR = LogisticRegression()
stratfit(clf_LR, train_data_os, train_labels_os, n_splits=5)
cv_scores_dict['5. Logistic Regression\nwith Oversampled Data'] = check_varian
ce(clf_LR

, train_data_os

, train_labels_os

, n_splits = 5

)
f1_dict['Logistic Regression with Oversampled Data'] = ['%.2f' % np.mean(accur
acy_temp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.52	0.72	0.60	8615
Died	0.50	0.73	0.60	8615
Euthanasia	0.53	0.37	0.44	8615
Return_to_owner	0.49	0.54	0.51	8615
Transfer	0.50	0.16	0.24	8615
micro avg	0.51	0.51	0.51	43075
macro avg	0.51	0.51	0.48	43075
weighted avg	0.51	0.51	0.48	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.51	0.71	0.60	2154
Died	0.51	0.74	0.60	2154
Euthanasia	0.52	0.37	0.43	2154
Return_to_owner	0.48	0.54	0.51	2154
Transfer	0.51	0.15	0.23	2154
micro avg	0.50	0.50	0.50	10770
macro avg	0.50	0.50	0.47	10770
weighted avg	0.50	0.50	0.47	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.52	0.73	0.61	8615
Died	0.51	0.74	0.60	8615
Euthanasia	0.52	0.37	0.43	8615
Return_to_owner	0.48	0.54	0.51	8615
Transfer	0.51	0.16	0.24	8615
micro avg	0.51	0.51	0.51	43075
macro avg	0.51	0.51	0.48	43075
weighted avg	0.51	0.51	0.48	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.52	0.71	0.60	2154
Died	0.50	0.72	0.59	2154
Euthanasia	0.52	0.37	0.44	2154
Return_to_owner	0.48	0.55	0.51	2154
Transfer	0.52	0.17	0.26	2154
micro avg	0.50	0.50	0.50	10770
macro avg	0.51	0.50	0.48	10770
weighted avg	0.51	0.50	0.48	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.52	0.72	0.60	8615
Died	0.50	0.72	0.59	8615
Euthanasia	0.53	0.37	0.44	8615
Return_to_owner	0.48	0.54	0.51	8615
Transfer	0.50	0.17	0.25	8615
micro avg	0.50	0.50	0.50	43075
macro avg	0.50	0.50	0.48	43075
weighted avg	0.50	0.50	0.48	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.52	0.71	0.60	2154
Died	0.51	0.73	0.60	2154
Euthanasia	0.50	0.36	0.42	2154
Return_to_owner	0.48	0.54	0.51	2154
Transfer	0.45	0.16	0.24	2154
micro avg	0.50	0.50	0.50	10770
macro avg	0.49	0.50	0.47	10770
weighted avg	0.49	0.50	0.47	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.52	0.72	0.60	8615
Died	0.51	0.75	0.61	8615
Euthanasia	0.53	0.37	0.44	8615
Return_to_owner	0.48	0.54	0.51	8615
Transfer	0.53	0.17	0.26	8615
micro avg	0.51	0.51	0.51	43075
macro avg	0.51	0.51	0.48	43075
weighted avg	0.51	0.51	0.48	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.51	0.72	0.60	2154
Died	0.50	0.74	0.60	2154
Euthanasia	0.52	0.38	0.44	2154
Return_to_owner	0.49	0.52	0.50	2154
Transfer	0.52	0.17	0.25	2154
micro avg	0.51	0.51	0.51	10770
macro avg	0.51	0.51	0.48	10770
weighted avg	0.51	0.51	0.48	10770

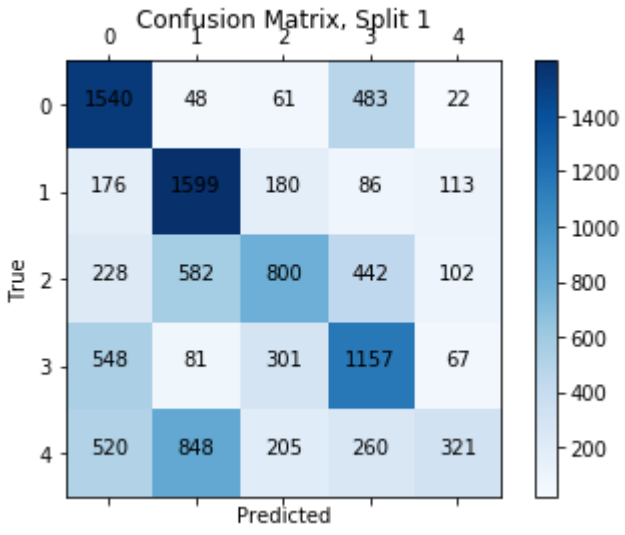
TRAIN: 43080 and TEST: 10765

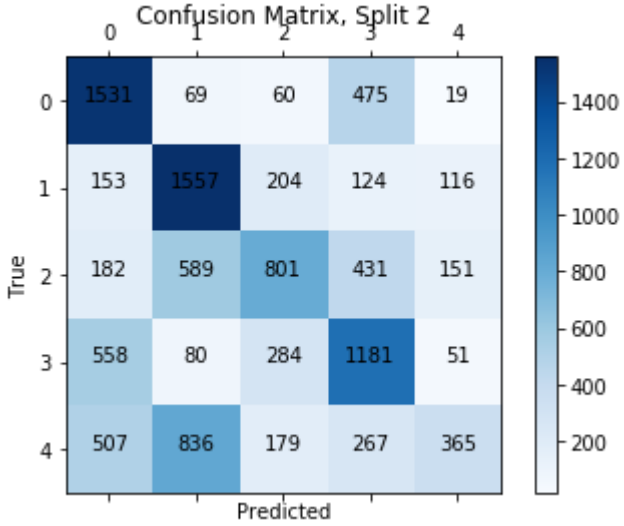
For train data:

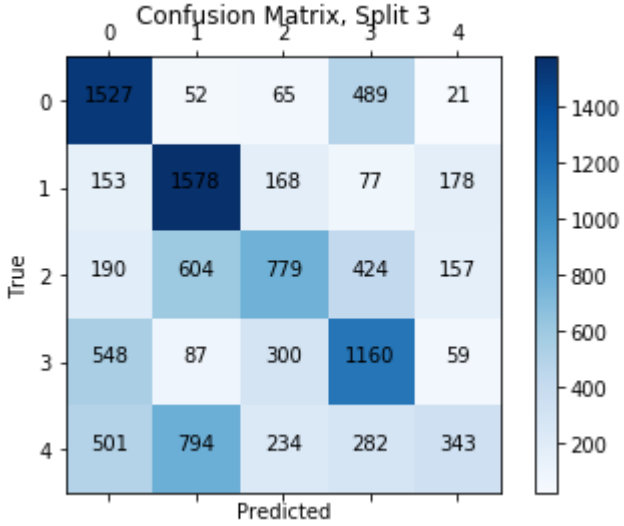
	precision	recall	f1-score	support
Adoption	0.52	0.72	0.60	8616
Died	0.50	0.73	0.60	8616
Euthanasia	0.53	0.38	0.44	8616
Return_to_owner	0.49	0.55	0.51	8616
Transfer	0.49	0.16	0.24	8616
micro avg	0.51	0.51	0.51	43080
macro avg	0.51	0.51	0.48	43080
weighted avg	0.51	0.51	0.48	43080

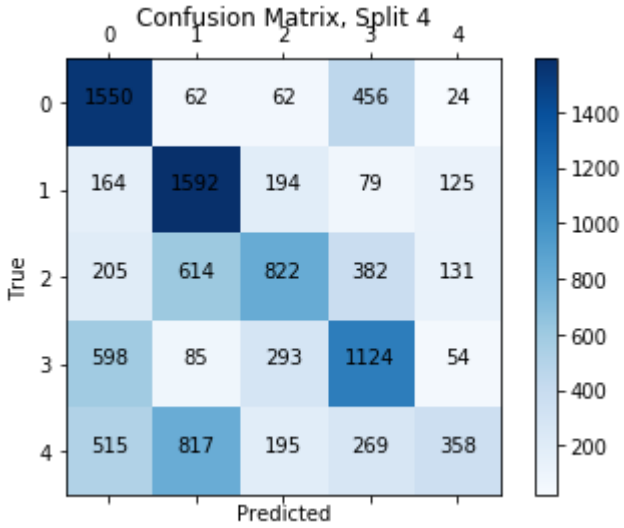
For test data:

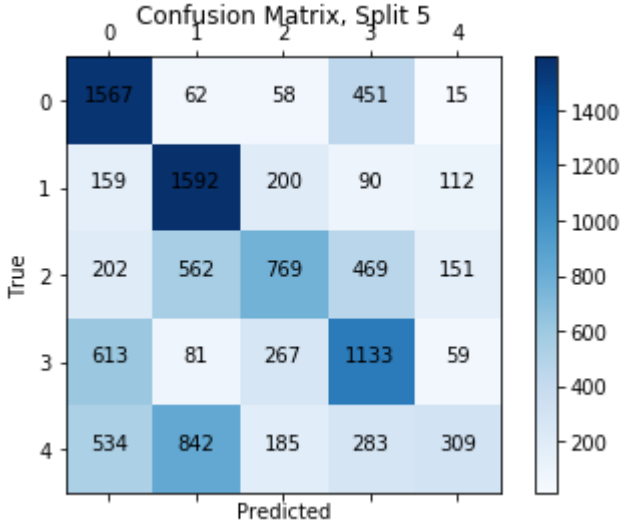
	precision	recall	f1-score	support
Adoption	0.51	0.73	0.60	2153
Died	0.51	0.74	0.60	2153
Euthanasia	0.52	0.36	0.42	2153
Return_to_owner	0.47	0.53	0.49	2153
Transfer	0.48	0.14	0.22	2153
micro avg	0.50	0.50	0.50	10765
macro avg	0.50	0.50	0.47	10765
weighted avg	0.50	0.50	0.47	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1531 69 60 475 19
1 153 1557 204 124 116
2 182 589 801 431 151
3 558 80 284 1181 51
4 507 836 179 267 365
True
Predicted
1400
1200
1000
800
600
400
200


Confusion Matrix, Split 3
0 1 2 3 4
0 1527 52 65 489 21
1 153 1578 168 77 178
2 190 604 779 424 157
3 548 87 300 1160 59
4 501 794 234 282 343
True
Predicted
1400
1200
1000
800
600
400
200


Confusion Matrix, Split 4
0 1 2 3 4
0 1550 62 62 456 24
1 164 1592 194 79 125
2 205 614 822 382 131
3 598 85 293 1124 54
4 515 817 195 269 358
True
Predicted
1400
1200
1000
800
600
400
200


Confusion Matrix, Split 5
0
1
2
3
4
True
Predicted
1567
62
58
451
15
159
1592
200
90
112
202
562
769
469
151
613
81
267
1133
59
534
842
185
283
309
1400
1200
1000
800
600
400
200

In [70]:
```
clf_RF = RandomForestClassifier()
stratfit(clf_RF, train_data_os, train_labels_os, n_splits=5)
cv_scores_dict['6. Random Forest\nwith Oversampled Data'] = check_variance(clf
_RF
                                                                          , t
rain_data_os
                                                                          , t
rain_labels_os
                                                                          , n
_splits = 5
                                                                         )
f1_dict['Random Forest with Oversampled Data'] = ['%.2f' % np.mean(accuracy_te
mp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.80	0.75	0.78	8615
Died	0.77	0.96	0.85	8615
Euthanasia	0.73	0.76	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.79	0.59	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.76	43075
weighted avg	0.77	0.77	0.76	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.68	0.62	0.65	2154
Died	0.77	0.95	0.85	2154
Euthanasia	0.71	0.75	0.73	2154
Return_to_owner	0.64	0.67	0.66	2154
Transfer	0.67	0.50	0.57	2154
micro avg	0.70	0.70	0.70	10770
macro avg	0.69	0.70	0.69	10770
weighted avg	0.69	0.70	0.69	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.76	0.77	8615
Died	0.77	0.96	0.85	8615
Euthanasia	0.73	0.77	0.75	8615
Return_to_owner	0.78	0.77	0.77	8615
Transfer	0.80	0.60	0.69	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.65	0.62	0.64	2154
Died	0.76	0.97	0.85	2154
Euthanasia	0.70	0.76	0.73	2154
Return_to_owner	0.65	0.66	0.66	2154
Transfer	0.66	0.45	0.53	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.68	10770
weighted avg	0.69	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.77	0.78	8615
Died	0.79	0.94	0.86	8615
Euthanasia	0.72	0.78	0.75	8615
Return_to_owner	0.77	0.76	0.77	8615
Transfer	0.79	0.60	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.67	0.62	0.65	2154
Died	0.77	0.93	0.85	2154
Euthanasia	0.66	0.76	0.71	2154
Return_to_owner	0.65	0.65	0.65	2154
Transfer	0.67	0.48	0.56	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.68	10770
weighted avg	0.69	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.80	0.75	0.77	8615
Died	0.79	0.95	0.86	8615
Euthanasia	0.72	0.78	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.79	0.61	0.69	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.66	0.64	0.65	2154
Died	0.79	0.93	0.85	2154
Euthanasia	0.68	0.75	0.71	2154
Return_to_owner	0.66	0.68	0.67	2154
Transfer	0.66	0.47	0.55	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.69	10770
weighted avg	0.69	0.69	0.69	10770

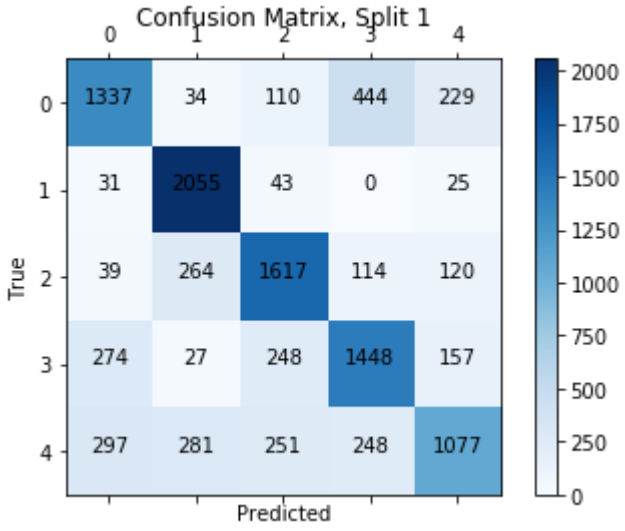
TRAIN: 43080 and TEST: 10765

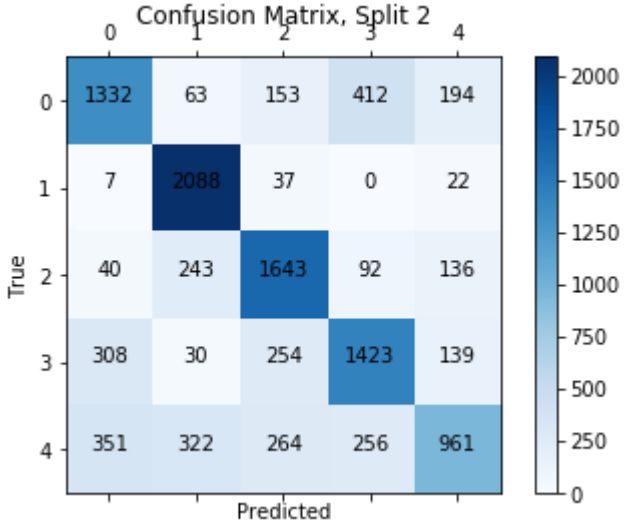
For train data:

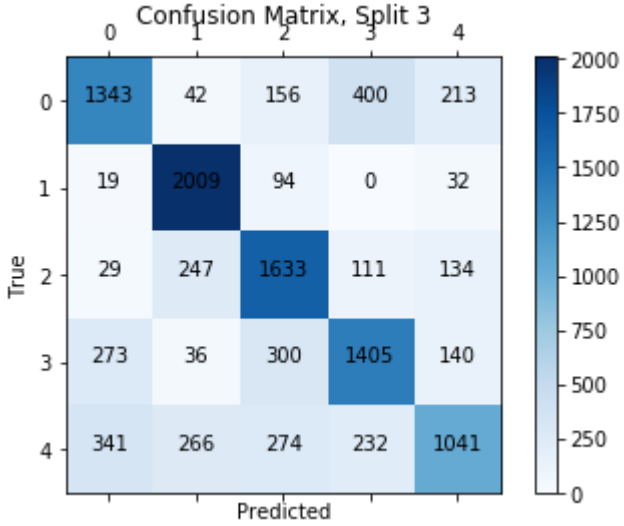
	precision	recall	f1-score	support
Adoption	0.79	0.76	0.78	8616
Died	0.78	0.94	0.86	8616
Euthanasia	0.73	0.77	0.75	8616
Return_to_owner	0.76	0.78	0.77	8616
Transfer	0.80	0.60	0.69	8616
micro avg	0.77	0.77	0.77	43080
macro avg	0.77	0.77	0.77	43080
weighted avg	0.77	0.77	0.77	43080

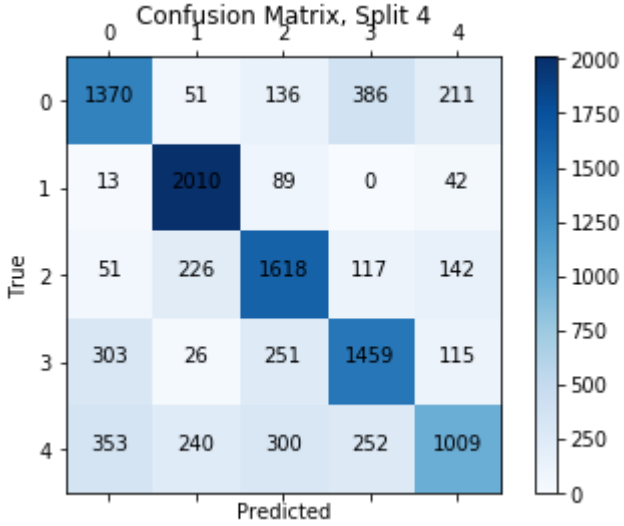
For test data:

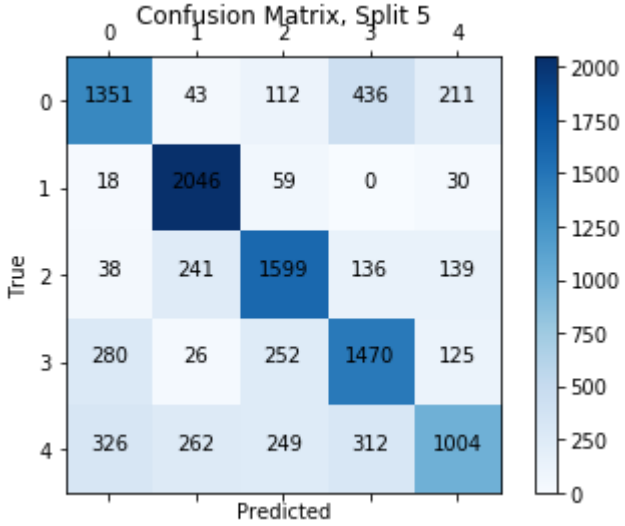
	precision	recall	f1-score	support
Adoption	0.67	0.63	0.65	2153
Died	0.78	0.95	0.86	2153
Euthanasia	0.70	0.74	0.72	2153
Return_to_owner	0.62	0.68	0.65	2153
Transfer	0.67	0.47	0.55	2153
micro avg	0.69	0.69	0.69	10765
macro avg	0.69	0.69	0.69	10765
weighted avg	0.69	0.69	0.69	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1332 63 153 412 194
1 7 2088 37 0 22
2 40 243 1643 92 136
3 308 30 254 1423 139
4 351 322 264 256 961
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 3
0 1 2 3 4
0 1343 42 156 400 213
1 19 2009 94 0 32
2 29 247 1633 111 134
3 273 36 300 1405 140
4 341 266 274 232 1041
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 4
0 1 2 3 4
0 1370 51 136 386 211
1 13 2010 89 0 42
2 51 226 1618 117 142
3 303 26 251 1459 115
4 353 240 300 252 1009
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 5
0
1
2
3
4
True
Predicted
1351
43
112
436
211
18
2046
59
0
30
38
241
1599
136
139
280
26
252
1470
125
326
262
249
312
1004
2000
1750
1500
1250
1000
750
500
250
0

In [71]:
```
clf_XGB = xgb.XGBClassifier()
stratfit(clf_XGB, train_data_os, train_labels_os, n_splits=5)
cv_scores_dict['7. XGBoost\nwith Oversampled Data'] = check_variance(clf_XGB
                                                                     , t
rain_data_os
                                                                     , t
rain_labels_os
                                                                     , n
_splits = 5
                                                                    )
f1_dict['XGBoost with Oversampled Data'] = ['%.2f' % np.mean(accuracy_temp_lis
t), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.59	0.64	0.62	8615
Died	0.62	0.68	0.65	8615
Euthanasia	0.52	0.44	0.48	8615
Return_to_owner	0.46	0.61	0.52	8615
Transfer	0.49	0.31	0.38	8615
micro avg	0.54	0.54	0.54	43075
macro avg	0.54	0.54	0.53	43075
weighted avg	0.54	0.54	0.53	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.59	0.65	0.62	2154
Died	0.61	0.68	0.64	2154
Euthanasia	0.50	0.42	0.46	2154
Return_to_owner	0.46	0.60	0.52	2154
Transfer	0.46	0.29	0.36	2154
micro avg	0.53	0.53	0.53	10770
macro avg	0.52	0.53	0.52	10770
weighted avg	0.52	0.53	0.52	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.59	0.65	0.62	8615
Died	0.61	0.69	0.65	8615
Euthanasia	0.52	0.44	0.48	8615
Return_to_owner	0.46	0.60	0.52	8615
Transfer	0.47	0.30	0.37	8615
micro avg	0.54	0.54	0.54	43075
macro avg	0.53	0.54	0.53	43075
weighted avg	0.53	0.54	0.53	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.59	0.66	0.62	2154
Died	0.62	0.70	0.66	2154
Euthanasia	0.51	0.43	0.47	2154
Return_to_owner	0.46	0.59	0.52	2154
Transfer	0.47	0.29	0.36	2154
micro avg	0.53	0.53	0.53	10770
macro avg	0.53	0.53	0.52	10770
weighted avg	0.53	0.53	0.52	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.59	0.64	0.62	8615
Died	0.62	0.69	0.65	8615
Euthanasia	0.52	0.44	0.48	8615
Return_to_owner	0.47	0.62	0.53	8615
Transfer	0.48	0.31	0.38	8615
micro avg	0.54	0.54	0.54	43075
macro avg	0.54	0.54	0.53	43075
weighted avg	0.54	0.54	0.53	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.60	0.65	0.62	2154
Died	0.62	0.69	0.65	2154
Euthanasia	0.53	0.45	0.49	2154
Return_to_owner	0.46	0.60	0.52	2154
Transfer	0.50	0.31	0.38	2154
micro avg	0.54	0.54	0.54	10770
macro avg	0.54	0.54	0.53	10770
weighted avg	0.54	0.54	0.53	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.60	0.64	0.62	8615
Died	0.61	0.70	0.65	8615
Euthanasia	0.52	0.43	0.47	8615
Return_to_owner	0.46	0.61	0.53	8615
Transfer	0.46	0.29	0.36	8615
micro avg	0.54	0.54	0.54	43075
macro avg	0.53	0.54	0.53	43075
weighted avg	0.53	0.54	0.53	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.61	0.65	0.63	2154
Died	0.63	0.70	0.66	2154
Euthanasia	0.52	0.43	0.47	2154
Return_to_owner	0.47	0.63	0.54	2154
Transfer	0.47	0.31	0.37	2154
micro avg	0.54	0.54	0.54	10770
macro avg	0.54	0.54	0.53	10770
weighted avg	0.54	0.54	0.53	10770

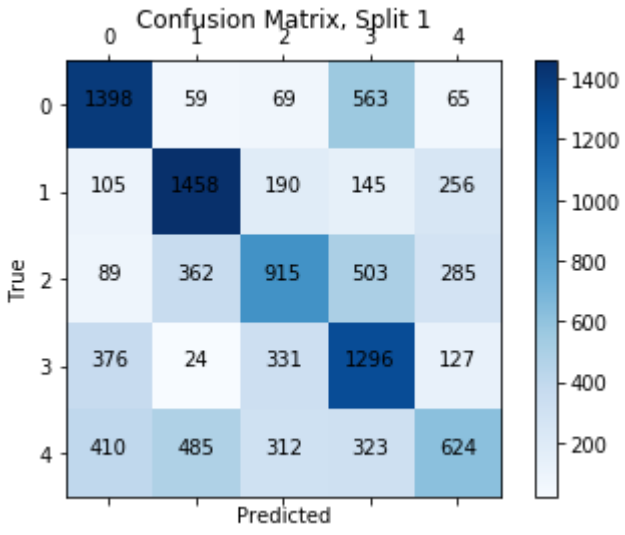
TRAIN: 43080 and TEST: 10765

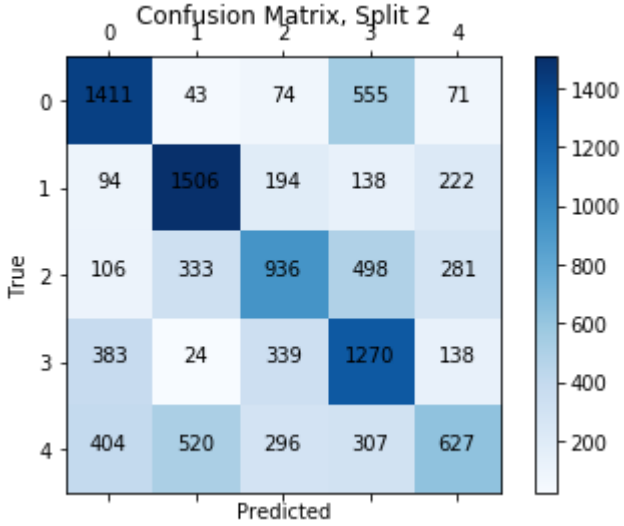
For train data:

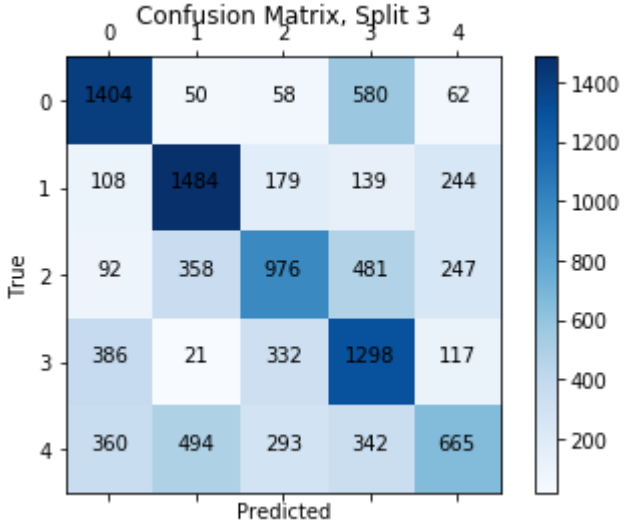
	precision	recall	f1-score	support
Adoption	0.60	0.66	0.63	8616
Died	0.62	0.68	0.65	8616
Euthanasia	0.52	0.43	0.47	8616
Return_to_owner	0.46	0.61	0.53	8616
Transfer	0.47	0.31	0.37	8616
micro avg	0.54	0.54	0.54	43080
macro avg	0.53	0.54	0.53	43080
weighted avg	0.53	0.54	0.53	43080

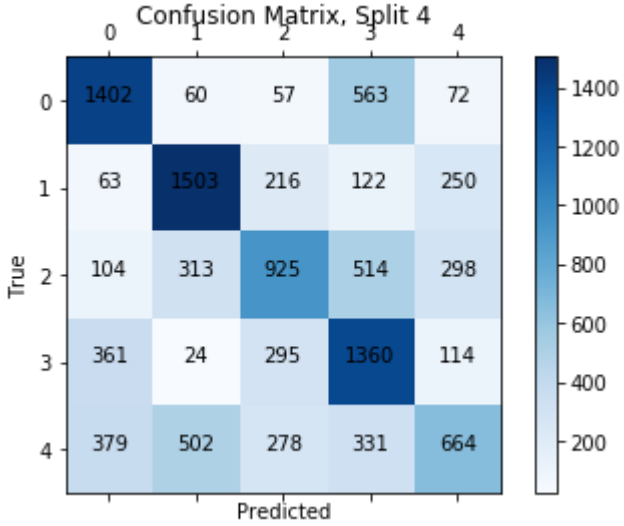
For test data:

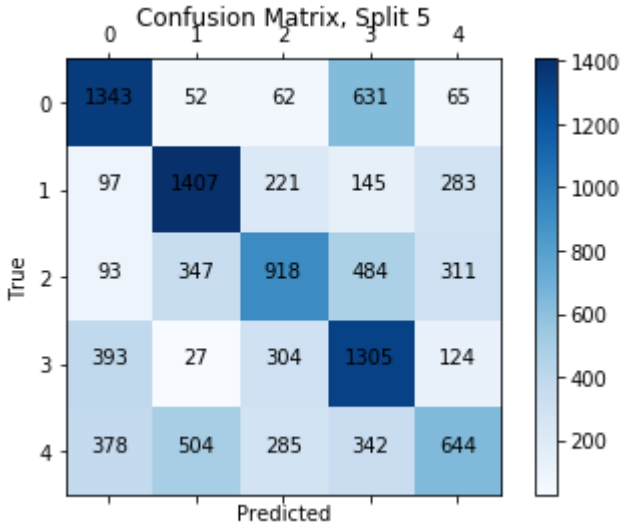
	precision	recall	f1-score	support
Adoption	0.58	0.62	0.60	2153
Died	0.60	0.65	0.63	2153
Euthanasia	0.51	0.43	0.47	2153
Return_to_owner	0.45	0.61	0.52	2153
Transfer	0.45	0.30	0.36	2153
micro avg	0.52	0.52	0.52	10765
macro avg	0.52	0.52	0.51	10765
weighted avg	0.52	0.52	0.51	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1411 43 74 555 71
1 94 1506 194 138 222
2 106 333 936 498 281
3 383 24 339 1270 138
4 404 520 296 307 627
True
Predicted
1400
1200
1000
800
600
400
200


Confusion Matrix, Split 3
0 1 2 3 4
0 1404 50 58 580 62
1 108 1484 179 139 244
2 92 358 976 481 247
3 386 21 332 1298 117
4 360 494 293 342 665
True
Predicted
1400
1200
1000
800
600
400
200


Confusion Matrix, Split 4
0 1 2 3 4
0 1402 60 57 563 72
1 63 1503 216 122 250
2 104 313 925 514 298
3 361 24 295 1360 114
4 379 502 278 331 664
True
Predicted
1400
1200
1000
800
600
400
200



Now, with the oversampled and balanced dataset, random forest returned a weighted f1-score of ~0.69 on the test folds, which is much better than before. We also see much better performance on died and euthanasia outcomes than with the unbalanced dataset. XGBoost performed substantially worse with the balanced dataset, returning a weighted f1-score of ~0.52. The logistic regression classifer also performed worse in this situation, returning a weighted f1-score of ~0.47.

The random forest classifer f1-score on the balanced dataset is the best that we have seen so far. In addition, the f1-scores of the training data folds and test data folds were also closer together within the random forest classifier, showing a lesser amount of generalization with unseen data. Therefore, moving forward, we will look exclusively at optimizing the random forest classifier.

It is possible that other oversampling techniques may do better. Next, we try the `SMOTE` oversampler to see how that performs.

In [72]:
```
# Oversample to balance outcome classes.
ros = SMOTE(random_state=0)
train_data_os2, train_labels_os2 = ros.fit_resample(train_data, train_labels)

# Count to double check how many are in each class.
print(sorted(Counter(train_labels_os2).items()))
```

```
[('Adoption', 10769), ('Died', 10769), ('Euthanasia', 10769), ('Return_to_own
er', 10769), ('Transfer', 10769)]
```

In [73]:
```
clf_RF = RandomForestClassifier()
stratfit(clf_RF, train_data_os2, train_labels_os2, n_splits=5)
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.82	0.80	8615
Died	0.91	0.98	0.94	8615
Euthanasia	0.84	0.86	0.85	8615
Return_to_owner	0.82	0.82	0.82	8615
Transfer	0.82	0.70	0.76	8615
micro avg	0.84	0.84	0.84	43075
macro avg	0.84	0.84	0.84	43075
weighted avg	0.84	0.84	0.84	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.68	0.65	2154
Died	0.86	0.93	0.89	2154
Euthanasia	0.69	0.69	0.69	2154
Return_to_owner	0.59	0.60	0.60	2154
Transfer	0.61	0.51	0.56	2154
micro avg	0.68	0.68	0.68	10770
macro avg	0.68	0.68	0.68	10770
weighted avg	0.68	0.68	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.82	0.80	8615
Died	0.92	0.97	0.94	8615
Euthanasia	0.84	0.86	0.85	8615
Return_to_owner	0.81	0.82	0.82	8615
Transfer	0.82	0.70	0.76	8615
micro avg	0.84	0.84	0.84	43075
macro avg	0.83	0.84	0.83	43075
weighted avg	0.83	0.84	0.83	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.66	0.64	2154
Died	0.87	0.93	0.90	2154
Euthanasia	0.71	0.72	0.71	2154
Return_to_owner	0.61	0.62	0.62	2154
Transfer	0.63	0.55	0.59	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.69	10770
weighted avg	0.69	0.69	0.69	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.82	0.80	8615
Died	0.91	0.97	0.94	8615
Euthanasia	0.85	0.85	0.85	8615
Return_to_owner	0.81	0.82	0.82	8615
Transfer	0.81	0.71	0.76	8615
micro avg	0.84	0.84	0.84	43075
macro avg	0.83	0.84	0.83	43075
weighted avg	0.83	0.84	0.83	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.68	0.66	2154
Died	0.87	0.93	0.90	2154
Euthanasia	0.71	0.70	0.71	2154
Return_to_owner	0.61	0.59	0.60	2154
Transfer	0.62	0.55	0.58	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.69	10770
weighted avg	0.69	0.69	0.69	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.82	0.80	8615
Died	0.91	0.98	0.94	8615
Euthanasia	0.85	0.84	0.85	8615
Return_to_owner	0.81	0.82	0.82	8615
Transfer	0.80	0.72	0.76	8615
micro avg	0.84	0.84	0.84	43075
macro avg	0.83	0.84	0.83	43075
weighted avg	0.83	0.84	0.83	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.63	0.68	0.65	2154
Died	0.88	0.93	0.90	2154
Euthanasia	0.72	0.69	0.70	2154
Return_to_owner	0.61	0.62	0.62	2154
Transfer	0.63	0.56	0.59	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.69	10770
weighted avg	0.69	0.69	0.69	10770

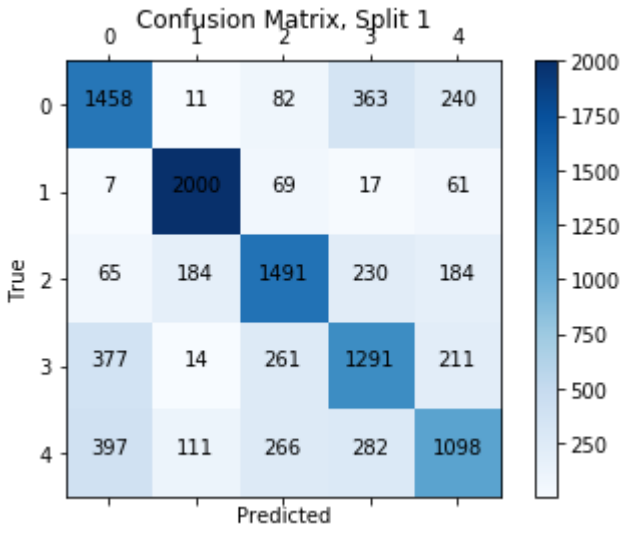
TRAIN: 43080 and TEST: 10765

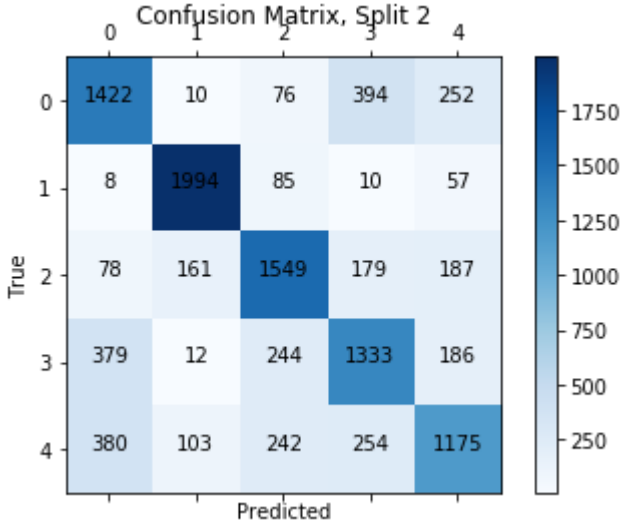
For train data:

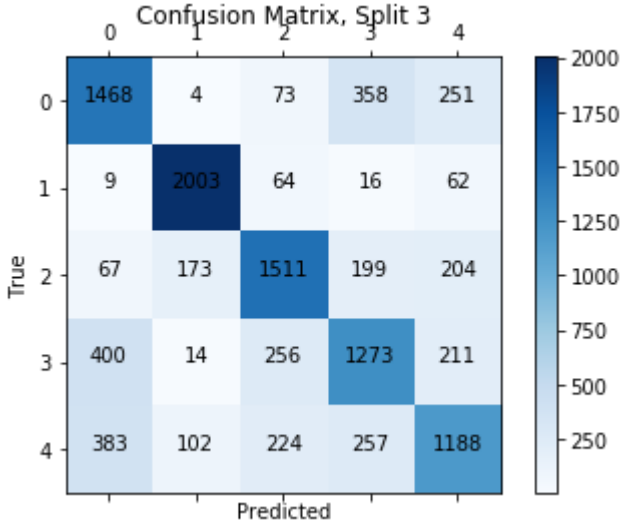
	precision	recall	f1-score	support
Adoption	0.80	0.82	0.81	8616
Died	0.91	0.98	0.94	8616
Euthanasia	0.87	0.84	0.85	8616
Return_to_owner	0.79	0.84	0.81	8616
Transfer	0.81	0.72	0.76	8616
micro avg	0.84	0.84	0.84	43080
macro avg	0.84	0.84	0.84	43080
weighted avg	0.84	0.84	0.84	43080

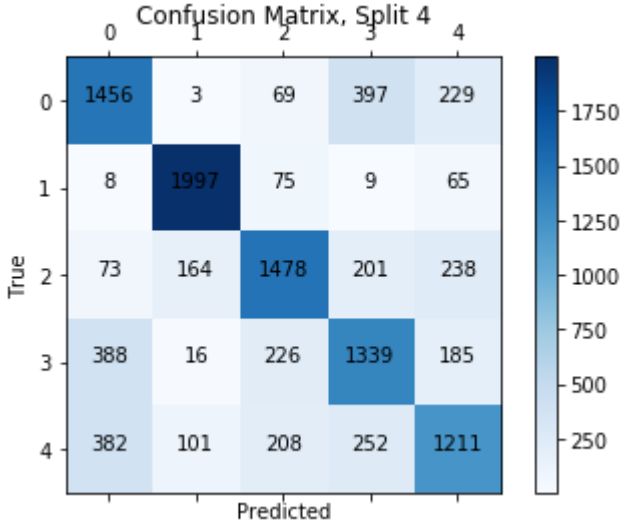
For test data:

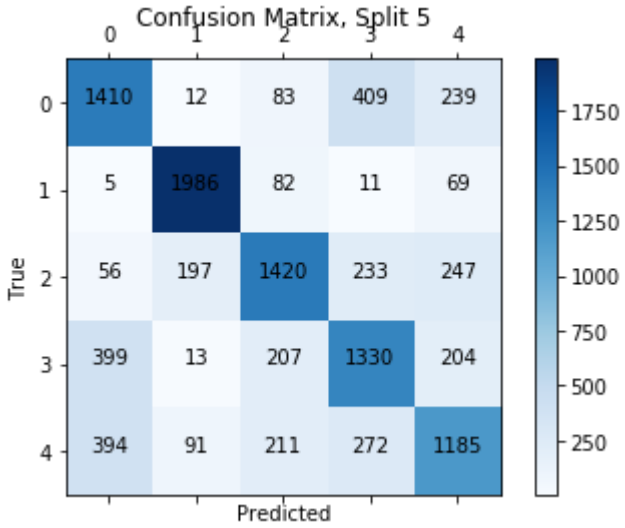
	precision	recall	f1-score	support
Adoption	0.62	0.65	0.64	2153
Died	0.86	0.92	0.89	2153
Euthanasia	0.71	0.66	0.68	2153
Return_to_owner	0.59	0.62	0.60	2153
Transfer	0.61	0.55	0.58	2153
micro avg	0.68	0.68	0.68	10765
macro avg	0.68	0.68	0.68	10765
weighted avg	0.68	0.68	0.68	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1422 10 76 394 252
1 8 1994 85 10 57
2 78 161 1549 179 187
3 379 12 244 1333 186
4 380 103 242 254 1175
True
Predicted
1750
1500
1250
1000
750
500
250


Confusion Matrix, Split 3
0 1 2 3 4
0 1468 4 73 358 251
1 9 2003 64 16 62
2 67 173 1511 199 204
3 400 14 256 1273 211
4 383 102 224 257 1188
True
Predicted
2000
1750
1500
1250
1000
750
500
250


Confusion Matrix, Split 4
0 1 2 3 4
0 1456 3 69 397 229
1 8 1997 75 9 65
2 73 164 1478 201 238
3 388 16 226 1339 185
4 382 101 208 252 1211
True
Predicted
1750
1500
1250
1000
750
500
250



It appears that the `SMOTE` oversampling technique not only takes a very long time to run, but also shows no performance improvements over the random oversampling case for the random forest classifier.

Next, we can perform a procedure of undersampling from the majority classes, just to see how well, or not, this performs. Undersampling takes the amount of examples present in the smallest class, and makes it so that each class can only have this many examples.

In [74]:
```
# Undersample to balance outcome classes.
rus = RandomUnderSampler(random_state=0)
train_data_us, train_labels_us = rus.fit_resample(train_data, train_labels)

print(sorted(Counter(train_labels_us).items()))
```

```
[('Adoption', 197), ('Died', 197), ('Euthanasia', 197), ('Return_to_owner', 1
97), ('Transfer', 197)]
```

In [75]:
```
# Running RF again, with undersampling
clf_RF = RandomForestClassifier()
stratfit(clf_RF, train_data_us, train_labels_us, n_splits=3)
f1_dict['Random Forest with Undersampled Data'] = ['%.2f' % np.mean(accuracy_t
emp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 655 and TEST: 330

For train data:

	precision	recall	f1-score	support
Adoption	0.92	0.93	0.92	131
Died	0.86	0.91	0.88	131
Euthanasia	0.90	0.87	0.89	131
Return_to_owner	0.93	0.95	0.94	131
Transfer	0.91	0.85	0.88	131
micro avg	0.90	0.90	0.90	655
macro avg	0.90	0.90	0.90	655
weighted avg	0.90	0.90	0.90	655

For test data:

	precision	recall	f1-score	support
Adoption	0.44	0.47	0.45	66
Died	0.43	0.48	0.45	66
Euthanasia	0.43	0.41	0.42	66
Return_to_owner	0.32	0.38	0.34	66
Transfer	0.21	0.14	0.17	66
micro avg	0.38	0.38	0.38	330
macro avg	0.36	0.38	0.37	330
weighted avg	0.36	0.38	0.37	330

TRAIN: 655 and TEST: 330

For train data:

	precision	recall	f1-score	support
Adoption	0.92	0.93	0.93	131
Died	0.86	0.91	0.88	131
Euthanasia	0.88	0.91	0.89	131
Return_to_owner	0.94	0.93	0.93	131
Transfer	0.93	0.85	0.89	131
micro avg	0.91	0.91	0.91	655
macro avg	0.91	0.91	0.91	655
weighted avg	0.91	0.91	0.91	655

For test data:

	precision	recall	f1-score	support
Adoption	0.45	0.50	0.47	66
Died	0.38	0.44	0.41	66
Euthanasia	0.41	0.39	0.40	66
Return_to_owner	0.43	0.41	0.42	66
Transfer	0.21	0.17	0.18	66
micro avg	0.38	0.38	0.38	330
macro avg	0.37	0.38	0.38	330
weighted avg	0.37	0.38	0.38	330

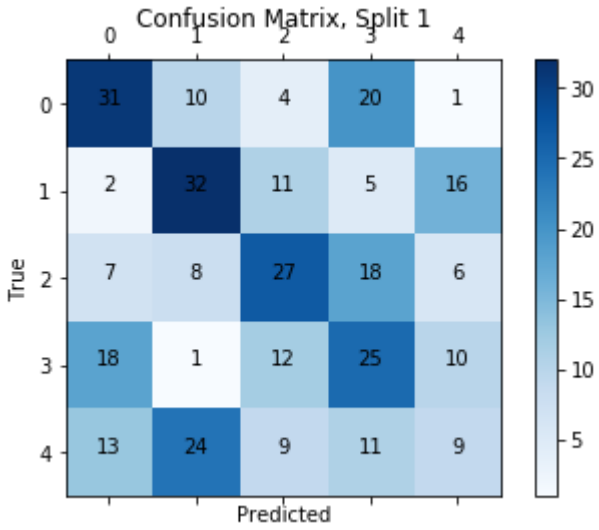
TRAIN: 660 and TEST: 325

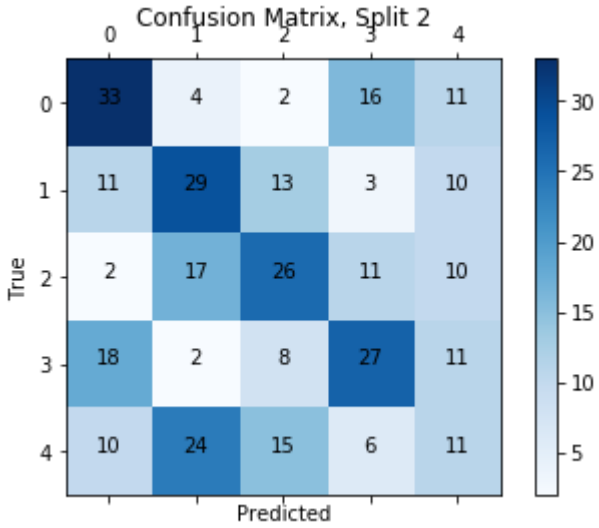
For train data:

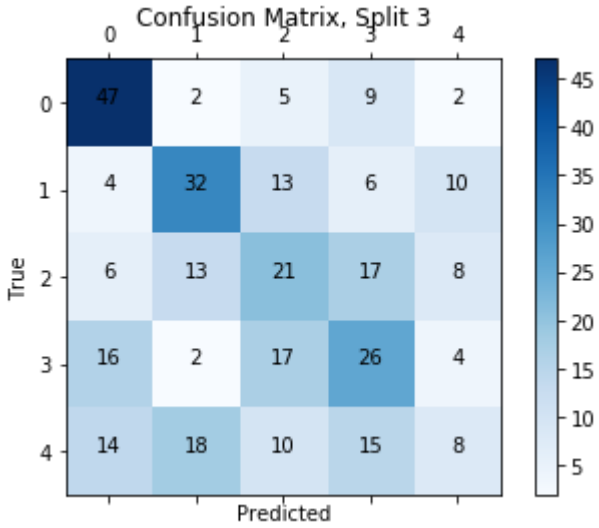
	precision	recall	f1-score	support
Adoption	0.87	0.94	0.91	132
Died	0.79	0.94	0.86	132
Euthanasia	0.88	0.86	0.87	132
Return_to_owner	0.94	0.92	0.93	132
Transfer	0.94	0.75	0.84	132
micro avg	0.88	0.88	0.88	660
macro avg	0.89	0.88	0.88	660
weighted avg	0.89	0.88	0.88	660

For test data:

	precision	recall	f1-score	support
Adoption	0.54	0.72	0.62	65
Died	0.48	0.49	0.48	65
Euthanasia	0.32	0.32	0.32	65
Return_to_owner	0.36	0.40	0.38	65
Transfer	0.25	0.12	0.16	65
micro avg	0.41	0.41	0.41	325
macro avg	0.39	0.41	0.39	325
weighted avg	0.39	0.41	0.39	325







Not surprisingly, we saw a huge generalization problem within the undersampled dataset. First, each class is limited to only 197 examples. The training folds also saw an almost perfect prediction with f1-scores at ~0.89, and test folds saw f1-scores of ~0.40. The training folds predicted very well, as there was so little data to learn, while the test folds did terribly, as the training data did not generalize well in such small quantities. We decided not to proceed with undersampling. Since random forest outperformed other classifiers by a substantial margin on the oversampled dataset, we opted to continue down the path of optimizing the performance of that approach.

Just to double check our intuition, we will test a Multinomial Naive Bayes classifier. We expect it to perform poorly, especially for such a sparsely populated dataset. We thought about testing the k Nearest Neighbors classifier as well, but given the sparsity of our features and the number of observations, we felt this would be not be worthwhile.

In [76]:

```
clf_MNB = MultinomialNB()

print('Oversampled:')
stratfit(clf_MNB, train_data_os, train_labels_os, n_splits=5)
f1_dict['Multinomial Naive Bayes with Oversampled Data'] = ['%.2f' % np.mean(a
ccuracy_temp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []

print('\nOriginal data:')
stratfit(clf_MNB, train_data, train_labels, n_splits=5)
f1_dict['Multinomial Naive Bayes with Raw Data'] = ['%.2f' % np.mean(accuracy_
temp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

Oversampled:

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.54	0.18	0.27	8615
Died	0.46	0.71	0.56	8615
Euthanasia	0.51	0.19	0.28	8615
Return_to_owner	0.37	0.75	0.50	8615
Transfer	0.26	0.18	0.21	8615
micro avg	0.40	0.40	0.40	43075
macro avg	0.43	0.40	0.36	43075
weighted avg	0.43	0.40	0.36	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.49	0.16	0.24	2154
Died	0.46	0.71	0.56	2154
Euthanasia	0.50	0.20	0.29	2154
Return_to_owner	0.36	0.74	0.49	2154
Transfer	0.25	0.18	0.21	2154
micro avg	0.40	0.40	0.40	10770
macro avg	0.41	0.40	0.36	10770
weighted avg	0.41	0.40	0.36	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.51	0.14	0.22	8615
Died	0.46	0.73	0.56	8615
Euthanasia	0.52	0.20	0.29	8615
Return_to_owner	0.37	0.75	0.50	8615
Transfer	0.24	0.17	0.20	8615
micro avg	0.40	0.40	0.40	43075
macro avg	0.42	0.40	0.35	43075
weighted avg	0.42	0.40	0.35	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.50	0.14	0.22	2154
Died	0.45	0.71	0.55	2154
Euthanasia	0.51	0.20	0.29	2154
Return_to_owner	0.37	0.76	0.50	2154
Transfer	0.25	0.18	0.21	2154
micro avg	0.40	0.40	0.40	10770
macro avg	0.42	0.40	0.35	10770

```
   weighted avg       0.42      0.40      0.35     10770


TRAIN: 43075 and TEST: 10770

For train data:
                  precision    recall  f1-score   support

       Adoption       0.53      0.13      0.21      8615
           Died       0.45      0.71      0.55      8615
     Euthanasia       0.51      0.19      0.28      8615
Return_to_owner       0.37      0.77      0.50      8615
       Transfer       0.25      0.19      0.22      8615

      micro avg       0.40      0.40      0.40     43075
      macro avg       0.42      0.40      0.35     43075
   weighted avg       0.42      0.40      0.35     43075

For test data:
                  precision    recall  f1-score   support

       Adoption       0.48      0.11      0.18      2154
           Died       0.46      0.73      0.56      2154
     Euthanasia       0.52      0.17      0.25      2154
Return_to_owner       0.37      0.77      0.50      2154
       Transfer       0.23      0.18      0.20      2154

      micro avg       0.39      0.39      0.39     10770
      macro avg       0.41      0.39      0.34     10770
   weighted avg       0.41      0.39      0.34     10770


TRAIN: 43075 and TEST: 10770

For train data:
                  precision    recall  f1-score   support

       Adoption       0.51      0.14      0.22      8615
           Died       0.46      0.71      0.56      8615
     Euthanasia       0.50      0.22      0.31      8615
Return_to_owner       0.37      0.73      0.50      8615
       Transfer       0.25      0.19      0.22      8615

      micro avg       0.40      0.40      0.40     43075
      macro avg       0.42      0.40      0.36     43075
   weighted avg       0.42      0.40      0.36     43075

For test data:
                  precision    recall  f1-score   support

       Adoption       0.49      0.12      0.19      2154
           Died       0.45      0.72      0.55      2154
     Euthanasia       0.47      0.22      0.30      2154
Return_to_owner       0.36      0.71      0.48      2154
       Transfer       0.24      0.18      0.20      2154

      micro avg       0.39      0.39      0.39     10770
```

```
      macro avg       0.40      0.39      0.35     10770
   weighted avg       0.40      0.39      0.35     10770


TRAIN: 43080 and TEST: 10765

For train data:
                  precision    recall  f1-score   support

       Adoption       0.55      0.17      0.26      8616
           Died       0.46      0.73      0.56      8616
     Euthanasia       0.46      0.22      0.30      8616
Return_to_owner       0.37      0.73      0.49      8616
       Transfer       0.25      0.17      0.20      8616

      micro avg       0.40      0.40      0.40     43080
      macro avg       0.42      0.40      0.36     43080
   weighted avg       0.42      0.40      0.36     43080

For test data:
                  precision    recall  f1-score   support

       Adoption       0.53      0.16      0.25      2153
           Died       0.46      0.74      0.56      2153
     Euthanasia       0.47      0.22      0.30      2153
Return_to_owner       0.37      0.72      0.49      2153
       Transfer       0.25      0.16      0.19      2153

      micro avg       0.40      0.40      0.40     10765
      macro avg       0.41      0.40      0.36     10765
   weighted avg       0.41      0.40      0.36     10765


Original data:

TRAIN: 21381 and TEST: 5348

For train data:
                  precision    recall  f1-score   support

       Adoption       0.54      0.65      0.59      8615
           Died       0.00      0.00      0.00       157
     Euthanasia       0.71      0.00      0.01      1244
Return_to_owner       0.41      0.38      0.39      3828
       Transfer       0.58      0.58      0.58      7537

      micro avg       0.53      0.53      0.53     21381
      macro avg       0.45      0.32      0.31     21381
   weighted avg       0.54      0.53      0.51     21381

For test data:
                  precision    recall  f1-score   support

       Adoption       0.52      0.63      0.57      2154
           Died       0.00      0.00      0.00        40
     Euthanasia       0.00      0.00      0.00       311
Return_to_owner       0.38      0.36      0.37       958
```

Transfer	0.56	0.54	0.55	1885
micro avg	0.51	0.51	0.51	5348
macro avg	0.29	0.31	0.30	5348
weighted avg	0.47	0.51	0.49	5348

TRAIN: 21382 and TEST: 5347

For train data:

	precision	recall	f1-score	support
Adoption	0.54	0.64	0.59	8615
Died	0.00	0.00	0.00	157
Euthanasia	0.38	0.00	0.00	1244
Return_to_owner	0.42	0.38	0.39	3829
Transfer	0.57	0.58	0.58	7537
micro avg	0.53	0.53	0.53	21382
macro avg	0.38	0.32	0.31	21382
weighted avg	0.51	0.53	0.51	21382

For test data:

	precision	recall	f1-score	support
Adoption	0.54	0.64	0.59	2154
Died	0.00	0.00	0.00	40
Euthanasia	0.33	0.00	0.01	311
Return_to_owner	0.38	0.34	0.36	957
Transfer	0.57	0.58	0.58	1885
micro avg	0.52	0.52	0.52	5347
macro avg	0.36	0.31	0.31	5347
weighted avg	0.50	0.52	0.50	5347

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.54	0.66	0.59	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.44	0.01	0.01	1244
Return_to_owner	0.41	0.35	0.38	3829
Transfer	0.57	0.57	0.57	7538
micro avg	0.53	0.53	0.53	21384
macro avg	0.39	0.32	0.31	21384
weighted avg	0.52	0.53	0.51	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.53	0.63	0.58	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.17	0.00	0.01	311

Return_to_owner	0.38	0.32	0.35	957
Transfer	0.57	0.60	0.58	1884
micro avg	0.52	0.52	0.52	5345
macro avg	0.33	0.31	0.30	5345
weighted avg	0.49	0.52	0.50	5345

TRAIN: 21384 and TEST: 5345

For train data:

	precision	recall	f1-score	support
Adoption	0.55	0.64	0.59	8615
Died	0.00	0.00	0.00	158
Euthanasia	0.45	0.00	0.01	1244
Return_to_owner	0.40	0.39	0.40	3829
Transfer	0.58	0.58	0.58	7538
micro avg	0.53	0.53	0.53	21384
macro avg	0.40	0.32	0.31	21384
weighted avg	0.52	0.53	0.51	21384

For test data:

	precision	recall	f1-score	support
Adoption	0.53	0.64	0.58	2154
Died	0.00	0.00	0.00	39
Euthanasia	0.00	0.00	0.00	311
Return_to_owner	0.38	0.36	0.37	957
Transfer	0.57	0.56	0.56	1884
micro avg	0.52	0.52	0.52	5345
macro avg	0.30	0.31	0.30	5345
weighted avg	0.48	0.52	0.50	5345

TRAIN: 21385 and TEST: 5344

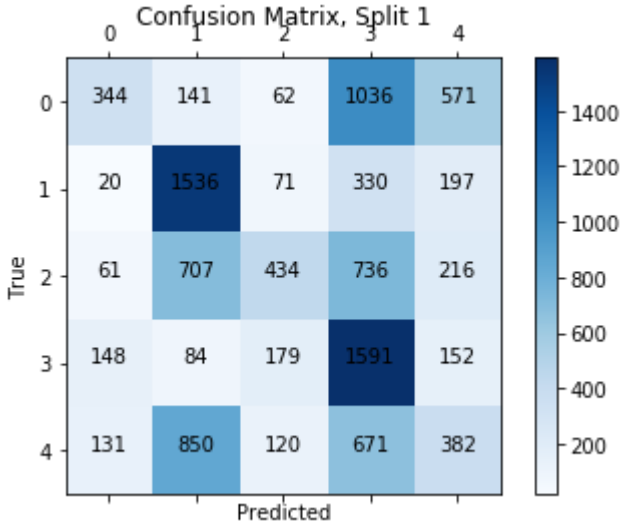
For train data:

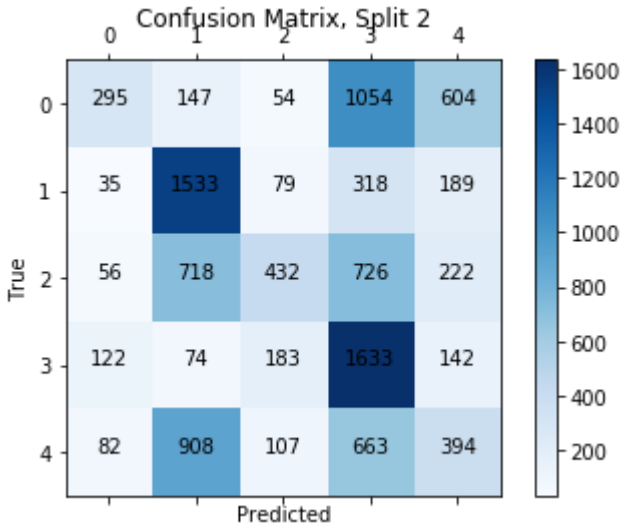
	precision	recall	f1-score	support
Adoption	0.54	0.65	0.59	8616
Died	0.00	0.00	0.00	158
Euthanasia	0.60	0.00	0.00	1244
Return_to_owner	0.40	0.38	0.39	3829
Transfer	0.58	0.57	0.57	7538
micro avg	0.53	0.53	0.53	21385
macro avg	0.42	0.32	0.31	21385
weighted avg	0.53	0.53	0.51	21385

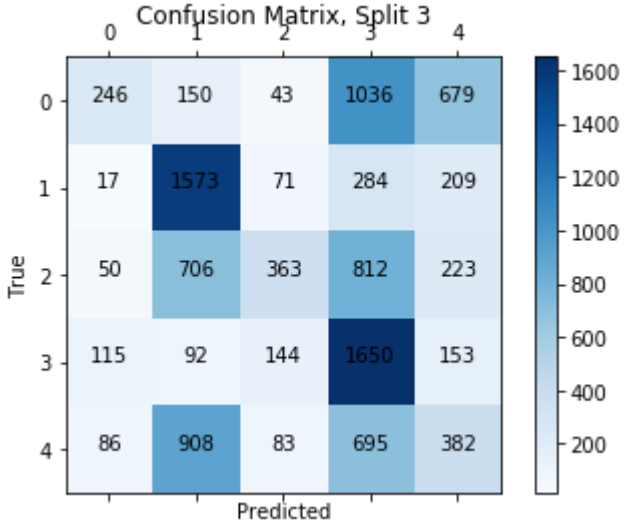
For test data:

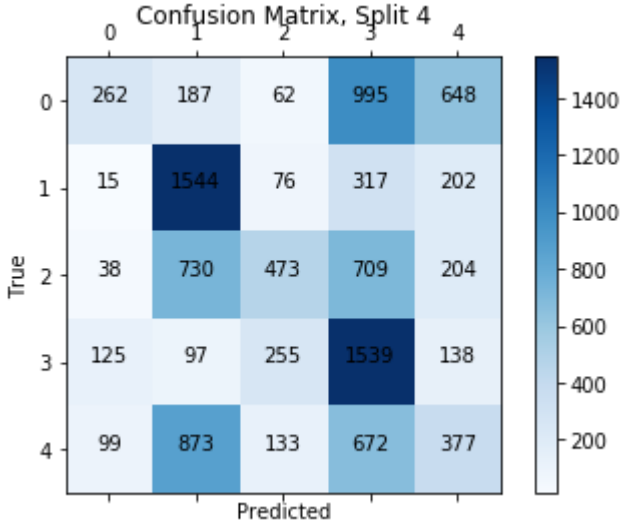
	precision	recall	f1-score	support
Adoption	0.52	0.61	0.56	2153
Died	0.00	0.00	0.00	39

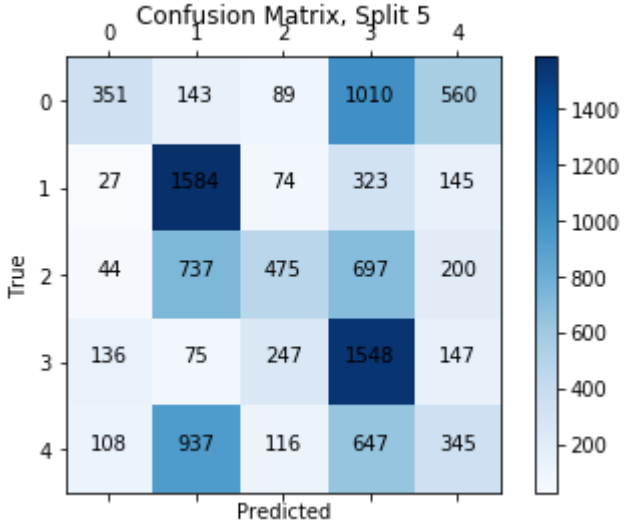
Euthanasia	0.00	0.00	0.00	311
Return_to_owner	0.36	0.37	0.36	957
Transfer	0.57	0.55	0.56	1884
micro avg	0.50	0.50	0.50	5344
macro avg	0.29	0.30	0.30	5344
weighted avg	0.47	0.50	0.49	5344

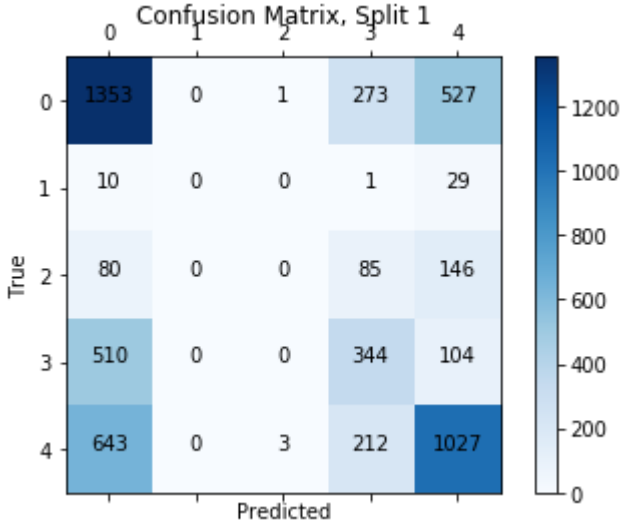


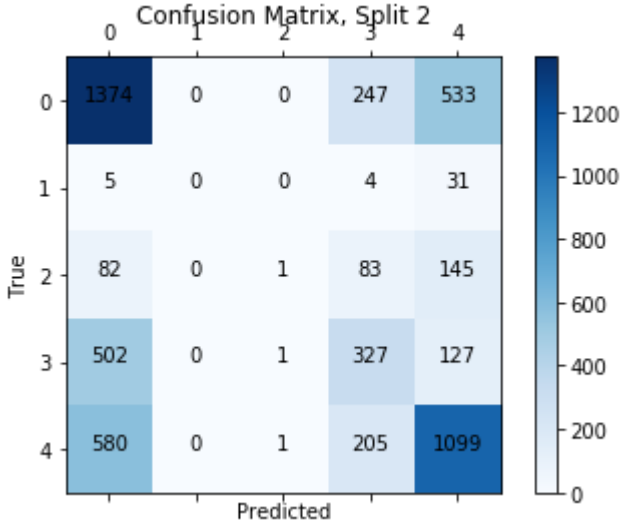


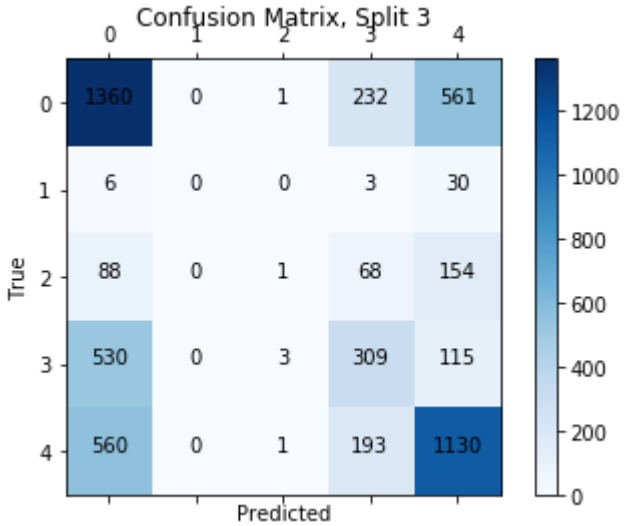

Confusion Matrix, Split 3
0 1 2 3 4
0 246 150 43 1036 679
1 17 1573 71 284 209
2 50 706 363 812 223
3 115 92 144 1650 153
4 86 908 83 695 382
True
Predicted
1600 1400 1200 1000 800 600 400 200

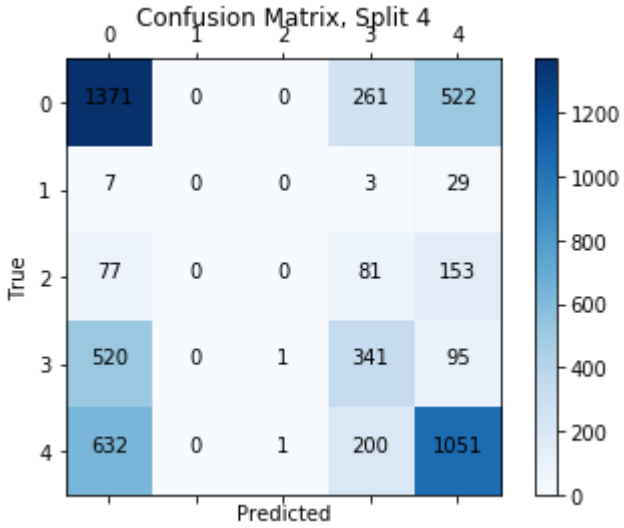

Confusion Matrix, Split 4
0 1 2 3 4
0 262 187 62 995 648
1 15 1544 76 317 202
2 38 730 473 709 204
3 125 97 255 1539 138
4 99 873 133 672 377
True
Predicted
1400 1200 1000 800 600 400 200

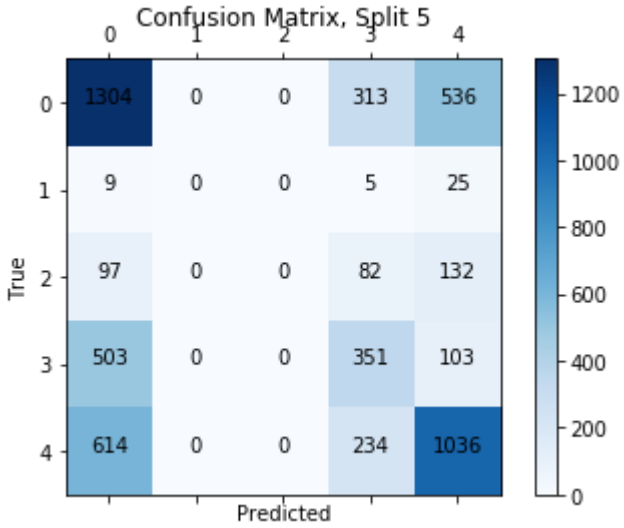

Confusion Matrix, Split 5
0 1 2 3 4
0 351 143 89 1010 560
1 27 1584 74 323 145
2 44 737 475 697 200
3 136 75 247 1548 147
4 108 937 116 647 345
True
Predicted
1400 1200 1000 800 600 400 200


Confusion Matrix, Split 1
0 1 2 3 4
0 1353 0 1 273 527
1 10 0 0 1 29
2 80 0 0 85 146
3 510 0 0 344 104
4 643 0 3 212 1027
True
Predicted
1200
1000
800
600
400
200
0


Confusion Matrix, Split 2
0 1 2 3 4
0 1374 0 0 247 533
1 5 0 0 4 31
2 82 0 1 83 145
3 502 0 1 327 127
4 580 0 1 205 1099
True
Predicted
1200
1000
800
600
400
200
0


Confusion Matrix, Split 3
0 1 2 3 4
0 1360 0 1 232 561
1 6 0 0 3 30
2 88 0 1 68 154
3 530 0 3 309 115
4 560 0 1 193 1130
True
Predicted
1200
1000
800
600
400
200
0





As expected, the Multinomial Naive Bayes classifier did not perform well, and was not worth exploring further. We are not surprised that the Multinomial Naive Bayes classifier did not perform well, because this classifier assumes feature independence, which we know is not the case in our dataset. As we saw from our EDA, the color and breed escriptions are often different for cats and dogs, speaking to feature dependence.

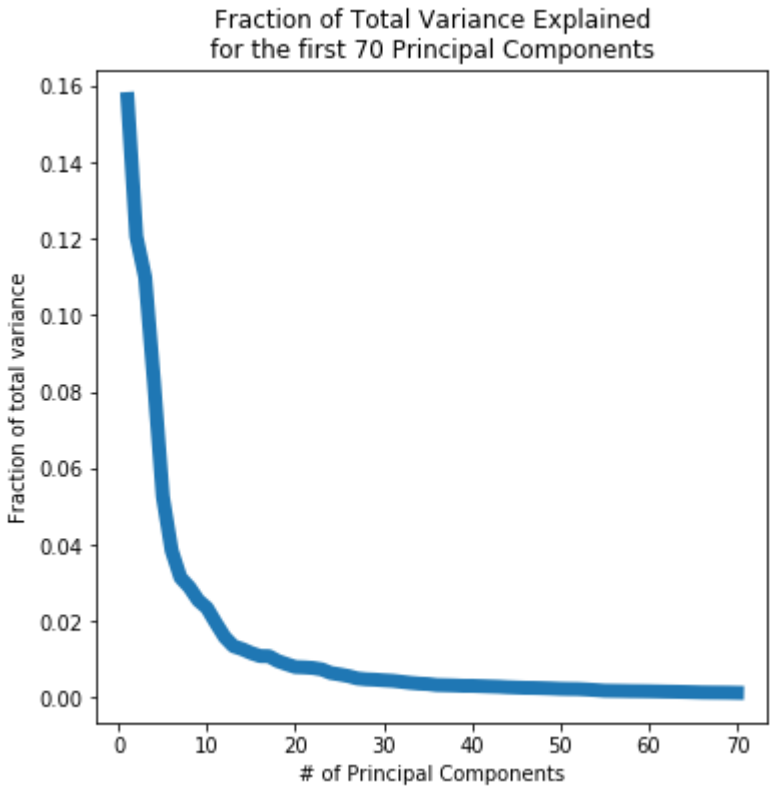
For the oversampled dataset, it returned a weighted f1-score of only ~0.35 for the test kfolds. For the original data, we see slightly better performance. However, we will not use Multinomial Naive Bayes, and are only leaving this in here as an example of a poorly performing model.

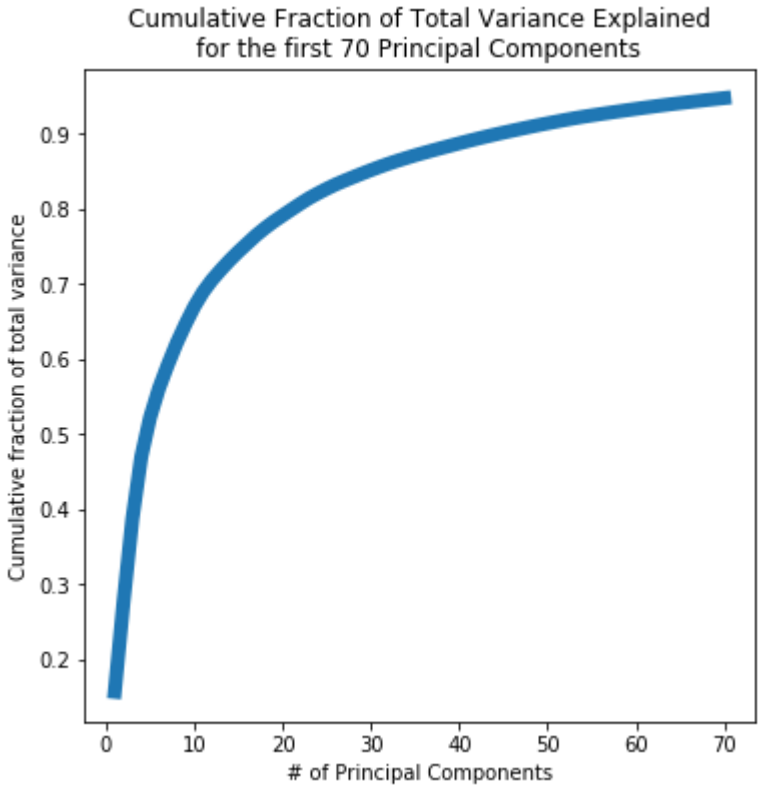
Principal Component Analysis

Our next step is to conduct principal component analysis (PCA) on our dataset. Our goal here is to see if we can reduce our features to their principal components without sacrificing too much of the explained variance. We still have over 200 breed variables, over 50 color variables, and several other features, some of which may not have been necessary, since our sparsity rate is over 94%. This method will likely speed up model fitting and simplify our model.

In [77]:

```
# Figure out best k to use - go up to 70.
k = 70

# Set the number of principal components.
pca = PCA(n_components=k)

# Fit principal component analysis.
pca.fit(train_data_os)

# Store explained variance ratio with corresponding k.
results = np.column_stack(([i for i in range(1, k+1)],pca.explained_variance_r
atio_))

# Plot the relationship of explained variance ratio.
fig1 = plt.figure(figsize = (6,6))
ax = fig1.add_subplot(111)
ax.set_title('Fraction of Total Variance Explained'
             '\nfor the first {} Principal Components'.format(k)
            )
plt.plot([i for i in range(1, k+1)],pca.explained_variance_ratio_, linewidth=
7.0)
ax.set_xlabel('# of Principal Components')
ax.set_ylabel('Fraction of total variance')
fig1.show()

# Plot the cumulative explained variance ratio.
fig2 = plt.figure(figsize = (6,6))
ax = fig2.add_subplot(111)
ax.set_title('Cumulative Fraction of Total Variance Explained'
             '\nfor the first {} Principal Components'.format(k)
            )
plt.plot([i for i in range(1, k+1)], np.cumsum(pca.explained_variance_ratio_),
linewidth=7.0)
ax.set_xlabel('# of Principal Components')
ax.set_ylabel('Cumulative fraction of total variance')
fig2.show()
```





When we look at the cumulative explained variance of 70 principal components, we see rather quickly that 30 principal components (our k) explained almost 90% of the variance within our dataset. We will now run a RandomForest classifier, our method of choice, with our PCA analysis taking 284 features down to 30 principal components.

In [78]:
```
# Set the number of principal components.
pca = PCA(n_components=30)

# Fit and transform based on train data, but oversampled.
projected_X_train_os = pca.fit_transform(train_data_os)
projected_X_train = pca.transform(train_data)

# Run PCA on test_data as well, so that we have a consistent number of feature
s.
projected_X_test = pca.transform(test_data)
```

In [79]:

```
clf_RF = RandomForestClassifier()
stratfit(clf_RF, projected_X_train_os, train_labels_os, n_splits=5)
cv_scores_dict['8. Random Forest\nwith Oversampled,\nProjected Data'] = check_
variance(clf_RF

, projected_X_train_os

, train_labels_os

, n_splits=5

)
f1_dict['Random Forest with Oversampled, Projected Data'] = ['%.2f' % np.mean(
accuracy_temp_list), '%.2f' % np.mean(f1_temp_list), '']
f1_temp_list = []
accuracy_temp_list = []
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.79	0.76	0.77	8615
Died	0.78	0.95	0.86	8615
Euthanasia	0.73	0.77	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.79	0.60	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.66	0.62	0.64	2154
Died	0.78	0.95	0.85	2154
Euthanasia	0.70	0.73	0.71	2154
Return_to_owner	0.65	0.67	0.66	2154
Transfer	0.64	0.48	0.55	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.68	0.69	0.68	10770
weighted avg	0.68	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.77	0.78	8615
Died	0.78	0.95	0.86	8615
Euthanasia	0.73	0.77	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.81	0.59	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.65	0.63	0.64	2154
Died	0.77	0.95	0.85	2154
Euthanasia	0.69	0.75	0.72	2154
Return_to_owner	0.64	0.65	0.65	2154
Transfer	0.66	0.46	0.54	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.68	0.69	0.68	10770
weighted avg	0.68	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.81	0.75	0.78	8615
Died	0.79	0.94	0.86	8615
Euthanasia	0.73	0.77	0.75	8615
Return_to_owner	0.75	0.78	0.77	8615
Transfer	0.79	0.61	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.67	0.60	0.63	2154
Died	0.78	0.94	0.86	2154
Euthanasia	0.69	0.75	0.72	2154
Return_to_owner	0.62	0.68	0.65	2154
Transfer	0.64	0.46	0.53	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.68	0.69	0.68	10770
weighted avg	0.68	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.77	0.78	8615
Died	0.78	0.94	0.86	8615
Euthanasia	0.73	0.77	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.79	0.60	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.77	43075
weighted avg	0.77	0.77	0.77	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.64	0.62	0.63	2154
Died	0.79	0.95	0.86	2154
Euthanasia	0.69	0.74	0.72	2154
Return_to_owner	0.64	0.67	0.66	2154
Transfer	0.66	0.46	0.54	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.68	0.69	0.68	10770
weighted avg	0.68	0.69	0.68	10770

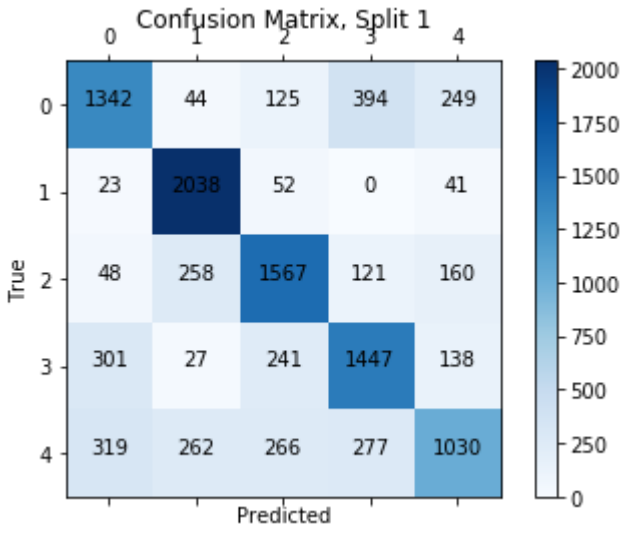
TRAIN: 43080 and TEST: 10765

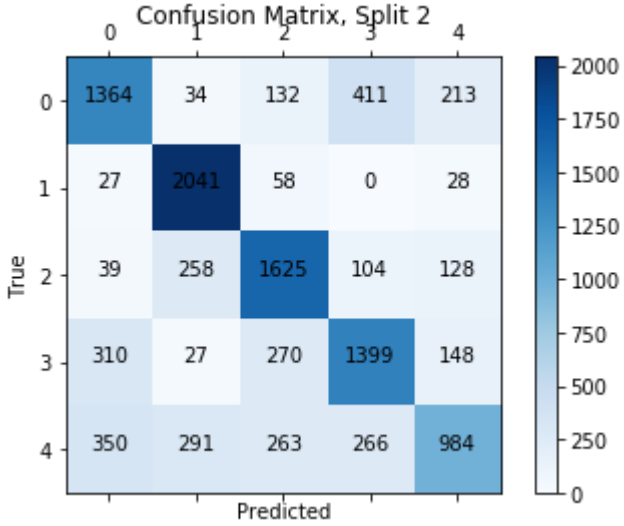
For train data:

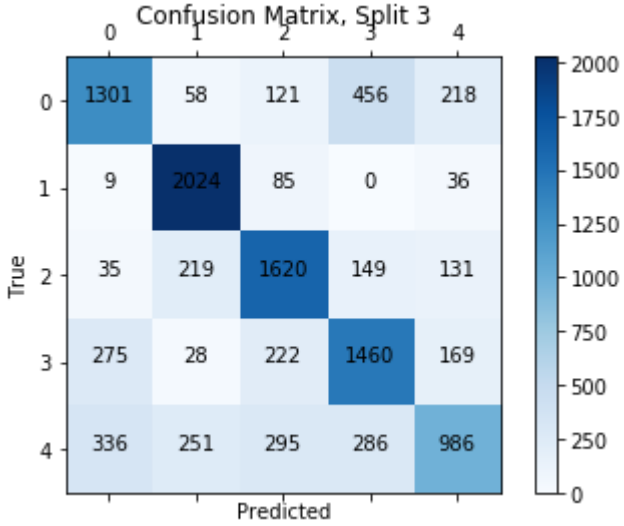
	precision	recall	f1-score	support
Adoption	0.78	0.77	0.78	8616
Died	0.78	0.95	0.86	8616
Euthanasia	0.72	0.78	0.75	8616
Return_to_owner	0.78	0.77	0.77	8616
Transfer	0.81	0.59	0.68	8616
micro avg	0.77	0.77	0.77	43080
macro avg	0.77	0.77	0.77	43080
weighted avg	0.77	0.77	0.77	43080

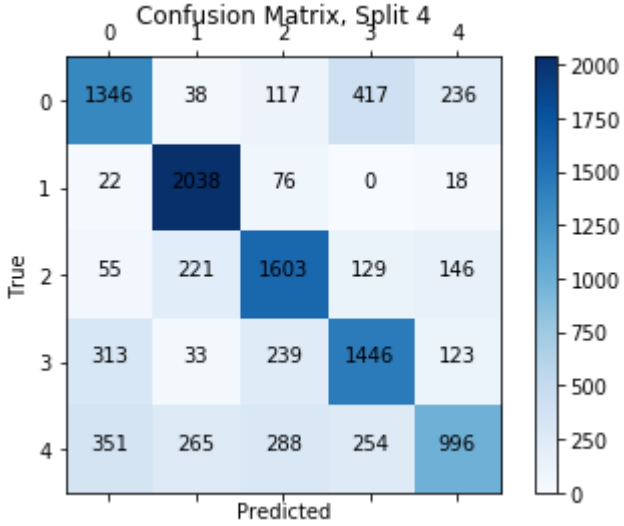
For test data:

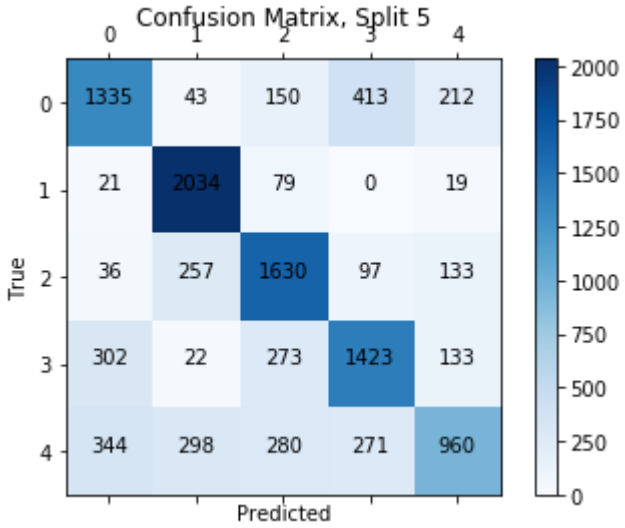
	precision	recall	f1-score	support
Adoption	0.66	0.62	0.64	2153
Died	0.77	0.94	0.85	2153
Euthanasia	0.68	0.76	0.71	2153
Return_to_owner	0.65	0.66	0.65	2153
Transfer	0.66	0.45	0.53	2153
micro avg	0.69	0.69	0.69	10765
macro avg	0.68	0.69	0.68	10765
weighted avg	0.68	0.69	0.68	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1364 34 132 411 213
1 27 2041 58 0 28
2 39 258 1625 104 128
3 310 27 270 1399 148
4 350 291 263 266 984
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 3
0 1 2 3 4
0 1301 58 121 456 218
1 9 2024 85 0 36
2 35 219 1620 149 131
3 275 28 222 1460 169
4 336 251 295 286 986
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 4
0 1 2 3 4
0 1346 38 117 417 236
1 22 2038 76 0 18
2 55 221 1603 129 146
3 313 33 239 1446 123
4 351 265 288 254 996
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0



While the weighted f1-score predictive performance did not improve on the random forest classifier with the addition of PCA, the model has a much faster run time, which would be important in a production environment. Given this finding, we choose to continue to optimize our algorithm with a random forest classifier on the oversampled PCA dataset.

Summary of Model F1-Scores

We have looked at many varieties of classifiers. Some are interesting, some are not. We have evaluated the differences in f1-scores. While these differences may seem obvious, we want to make sure we do hypothesis tests to check if they are, indeed, statistically significantly different. In order to do this, we created a `cv_scores_dict` to store the f1-scores from 5 cross-validation fits for each classifier. We convert this to a pandas data frame below.

In [80]:
```
# Create pandas dataframe with all f1-scores for all classifiers of interest.
cv_scores_df = pd.DataFrame.from_dict(cv_scores_dict)
print('Weighted F1-Score Distributions')
cv_scores_df
```

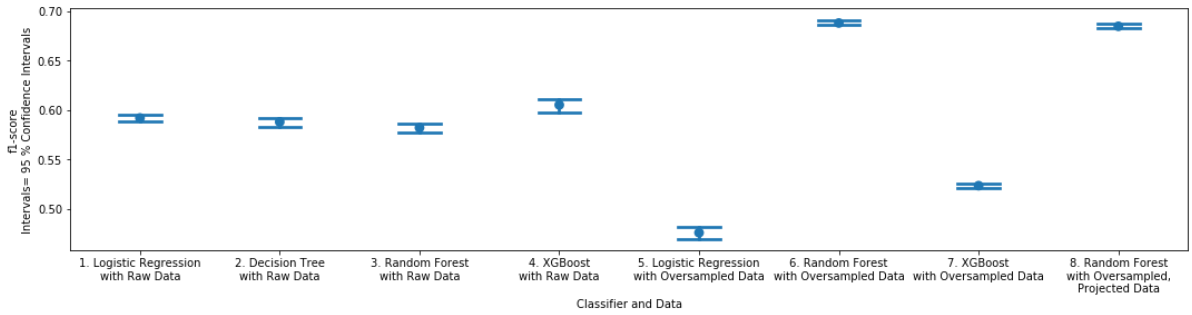
```
Weighted F1-Score Distributions
```

Out[80]:

	1. Logistic Regression with Raw Data	2. Decision Tree with Raw Data	3. Random Forest with Raw Data	4. XGBoost with Raw Data	5. Logistic Regression with Oversampled Data	6. Random Forest with Oversampled Data	7. XGBoost with Oversampled Data	8. Ra Fores Oversar Pro
0	0.591879	0.587784	0.580303	0.602546	0.471428	0.686789	0.527570	0.6
1	0.595342	0.588425	0.590345	0.612773	0.480132	0.685224	0.520553	0.6
2	0.591042	0.593996	0.575699	0.610277	0.478297	0.686816	0.524376	0.6
3	0.584260	0.590098	0.578653	0.590680	0.465702	0.693877	0.524412	0.6
4	0.596984	0.579116	0.585236	0.609197	0.484045	0.686657	0.520381	0.6

Now, we would like to plot the mean f1-score for each of the classifier/data combinations we have tested above. In order to test if the f1-scores are statistically significantly different from each other, we can do two-sample t-tests for pairs of classifiers. Alternatively, we can evaluate the 95% confidence intervals for the f1-scores for the classifier/data combinations. We plot this 95% confidence interval below. The f1-scores are significantly different in all cases in which confidence intervals do not overlap.

In [81]:
```
# Plot interval plot of 95 % confidence intervals for f1 scores for all intere
sting classifiers from above.
plt.figure(figsize=(18, 4))
ax = sns.pointplot(data=cv_scores_df, join=False, capsize=.3, s=.2)
ax.set_xticklabels(ax.get_xticklabels())
plt.xlabel("Classifier and Data")
plt.ylabel("f1-score\nIntervals= 95 % Confidence Intervals")
plt.show()
```



We can see here that the f1-scores for the classifiers fit with raw data are significantly lower than the random forest classifiers fit with oversampled data. The two random forest classifiers fit with oversampled data or oversampled and projected data and the XGBoost classifier do not show any difference in f1-scores. Because of this, and the faster speed of fitting the PCA projected data, **we choose the random forest classifier fit with oversampled, projected data to optimize further**.

Optimize the Best Algorithm (Random Forest)

Now, we optimize the random forest. We will select more parameters to vary and use `GridSearchCV` to make sure we are happy with the performance.

We will perform a GridSearch on our random forest classifier to identify its best parameters. We want to fine tune the parameters max_depth, min_samples_split, the criterion, and the n_estimators in order to improve the model's ability to generalize.

In [82]:
```
# This code optimizes a random forest classifier.
clf = RandomForestClassifier()

param = {'max_depth': np.arange(20,120,20)
         , 'min_samples_split': np.arange(4,10,2)
         # , 'n_estimators': np.arange(10, 120, 30); decided not to pursue for
speed
         , 'criterion': ['gini', 'entropy']}

# Optimize classifier, use f1_weighting, use 5 StratifiedKFolds.
clf_best_param = GridSearchCV(clf, param_grid = param, cv=5, scoring = 'f1_wei
ghted')

# Fit optimized classifier.
clf_best = clf_best_param.fit(projected_X_train_os, train_labels_os)

# Predict values on regular data.
pred_labels = clf_best_param.predict(projected_X_train)

# Print best classifier.
print(clf_best_param)

# Print classification report.
print(classification_report(train_labels, pred_labels))
```

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=RandomForestClassifier(bootstrap=True, class_weight=None, cr
iterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators='warn', n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'max_depth': array([ 20,  40,  60,  80, 100]), 'min_sample
s_split': array([4, 6, 8]), 'criterion': ['gini', 'entropy']},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring='f1_weighted', verbose=0)
                  precision    recall  f1-score   support

       Adoption       0.81      0.76      0.78     10769
           Died       0.11      0.82      0.19       197
     Euthanasia       0.38      0.69      0.49      1555
Return_to_owner       0.59      0.71      0.65      4786
       Transfer       0.85      0.59      0.69      9422

      micro avg       0.69      0.69      0.69     26729
      macro avg       0.55      0.71      0.56     26729
   weighted avg       0.75      0.69      0.70     26729
```

In [83]:
```
print(clf_best.best_params_)
```

```
{'criterion': 'gini', 'max_depth': 100, 'min_samples_split': 4}
```

Changing the number of estimators (number of trees in each decision forest) took a very long time in conjunction with all of the other parameters. We tried the above classifier with n_estimators=120, and saw no improvement over the default of n_estimators=10. In the interest of speed, we eliminated the n_estimators from our `GridSearchCV`.

Next, we run the best model output by the `GridSearchCV`.

In [84]:
```
# Run random forest with the best parameters.
clf_RF = RandomForestClassifier(max_depth=40, min_samples_split=4, criterion=
'gini')
stratfit(clf_RF, projected_X_train_os, train_labels_os, n_splits=5)
```

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.80	0.74	0.77	8615
Died	0.79	0.94	0.86	8615
Euthanasia	0.72	0.77	0.74	8615
Return_to_owner	0.76	0.77	0.77	8615
Transfer	0.77	0.61	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.76	43075
weighted avg	0.77	0.77	0.76	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.67	0.62	0.65	2154
Died	0.79	0.94	0.86	2154
Euthanasia	0.69	0.77	0.73	2154
Return_to_owner	0.64	0.67	0.65	2154
Transfer	0.65	0.48	0.55	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.69	10770
weighted avg	0.69	0.69	0.69	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.76	0.77	8615
Died	0.79	0.94	0.86	8615
Euthanasia	0.73	0.78	0.75	8615
Return_to_owner	0.77	0.77	0.77	8615
Transfer	0.78	0.60	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.76	43075
weighted avg	0.77	0.77	0.76	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.66	0.63	0.64	2154
Died	0.79	0.94	0.86	2154
Euthanasia	0.68	0.74	0.71	2154
Return_to_owner	0.65	0.67	0.66	2154
Transfer	0.66	0.47	0.55	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.68	10770
weighted avg	0.69	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.80	0.75	0.77	8615
Died	0.76	0.97	0.85	8615
Euthanasia	0.74	0.76	0.75	8615
Return_to_owner	0.75	0.78	0.77	8615
Transfer	0.81	0.58	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.76	43075
weighted avg	0.77	0.77	0.76	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.66	0.62	0.64	2154
Died	0.75	0.97	0.85	2154
Euthanasia	0.71	0.72	0.71	2154
Return_to_owner	0.63	0.68	0.65	2154
Transfer	0.68	0.45	0.54	2154
micro avg	0.69	0.69	0.69	10770
macro avg	0.69	0.69	0.68	10770
weighted avg	0.69	0.69	0.68	10770

TRAIN: 43075 and TEST: 10770

For train data:

	precision	recall	f1-score	support
Adoption	0.78	0.76	0.77	8615
Died	0.78	0.95	0.86	8615
Euthanasia	0.74	0.76	0.75	8615
Return_to_owner	0.76	0.78	0.77	8615
Transfer	0.79	0.59	0.68	8615
micro avg	0.77	0.77	0.77	43075
macro avg	0.77	0.77	0.76	43075
weighted avg	0.77	0.77	0.76	43075

For test data:

	precision	recall	f1-score	support
Adoption	0.65	0.61	0.63	2154
Died	0.76	0.95	0.85	2154
Euthanasia	0.69	0.72	0.71	2154
Return_to_owner	0.63	0.68	0.65	2154
Transfer	0.65	0.46	0.54	2154
micro avg	0.68	0.68	0.68	10770
macro avg	0.68	0.68	0.68	10770
weighted avg	0.68	0.68	0.68	10770

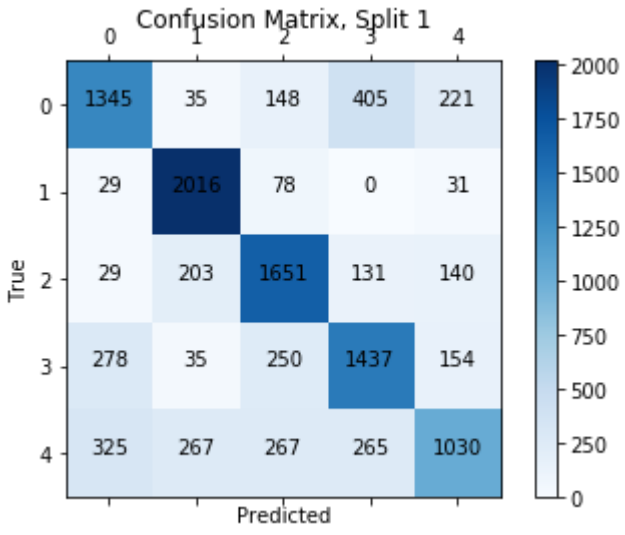
TRAIN: 43080 and TEST: 10765

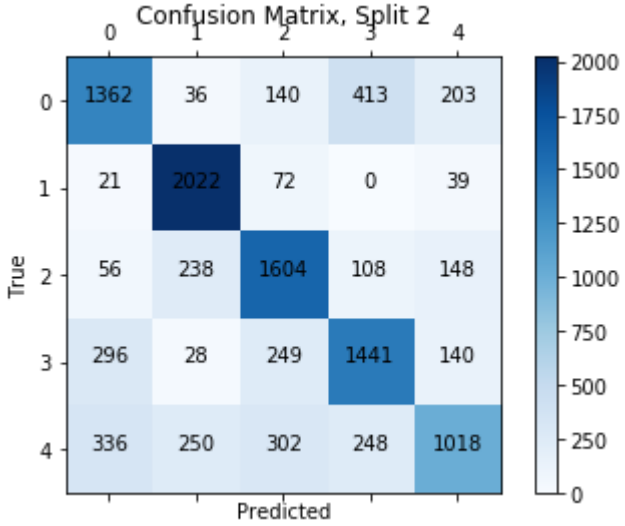
For train data:

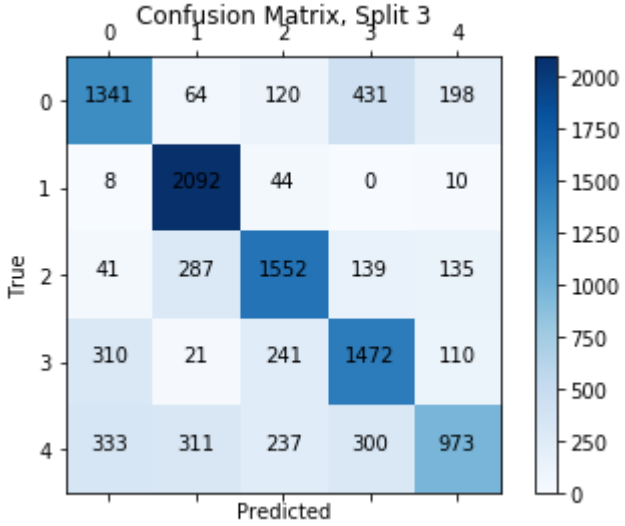
	precision	recall	f1-score	support
Adoption	0.79	0.76	0.78	8616
Died	0.78	0.94	0.86	8616
Euthanasia	0.73	0.77	0.75	8616
Return_to_owner	0.77	0.76	0.77	8616
Transfer	0.77	0.61	0.68	8616
micro avg	0.77	0.77	0.77	43080
macro avg	0.77	0.77	0.77	43080
weighted avg	0.77	0.77	0.77	43080

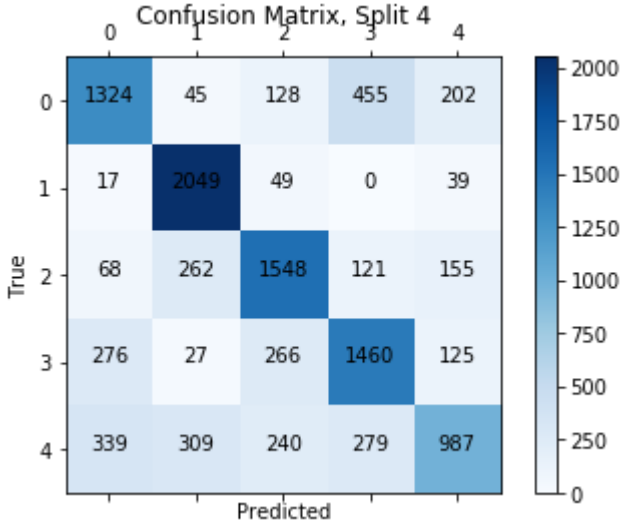
For test data:

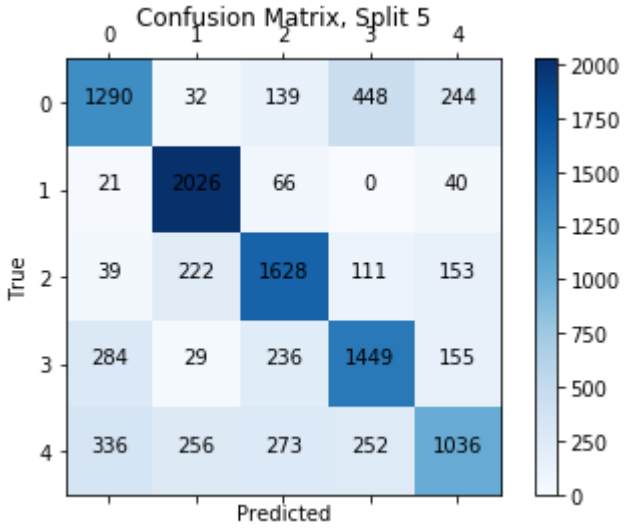
	precision	recall	f1-score	support
Adoption	0.65	0.60	0.63	2153
Died	0.79	0.94	0.86	2153
Euthanasia	0.70	0.76	0.72	2153
Return_to_owner	0.64	0.67	0.66	2153
Transfer	0.64	0.48	0.55	2153
micro avg	0.69	0.69	0.69	10765
macro avg	0.68	0.69	0.68	10765
weighted avg	0.68	0.69	0.68	10765




Confusion Matrix, Split 2
0 1 2 3 4
0 1362 36 140 413 203
1 21 2022 72 0 39
2 56 238 1604 108 148
3 296 28 249 1441 140
4 336 250 302 248 1018
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 3
0 1 2 3 4
0 1341 64 120 431 198
1 8 2092 44 0 10
2 41 287 1552 139 135
3 310 21 241 1472 110
4 333 311 237 300 973
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0


Confusion Matrix, Split 4
0 1 2 3 4
0 1324 45 128 455 202
1 17 2049 49 0 39
2 68 262 1548 121 155
3 276 27 266 1460 125
4 339 309 240 279 987
True
Predicted
2000 1750 1500 1250 1000 750 500 250 0



We found that the best parameters included a max_depth of 40, minimum_sample_split of 4, and criterion gini. The max_depth represents the maximum depth of each decision tree, or the maximum number of levels of each of those trees. The min_samples_split is the minimum number of observations or samples placed in a node before the node is split. We had to evaluate a balance with min_samples_split, as one data point per node would not help us generalize to test data, but each node could be more constrained as more samples have to be included. The same rule of balance goes for max_depth, as too many levels would make it hard for our algorithm to generalize on training data. Limiting the depth of the trees in our random forest helps reduce the number of important features. The criterion in this instance refers to the quality of each node split within our tree. The gini criterion was chosen as the best parameter over the entropy criterion. This measures how often a randomly chosen observation within our training data would be mislabeled.

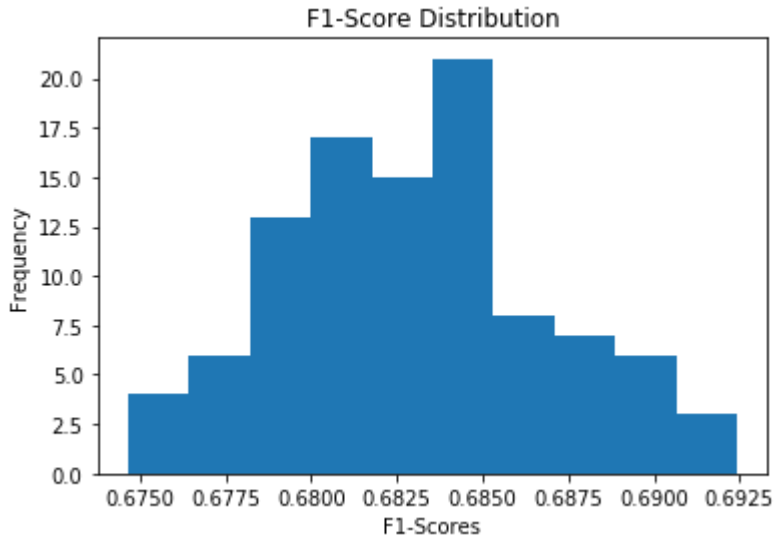
Overall, the random forest classifier improved less than 1 percentage point on the weighted f1-score with the addition of PCA and the optimized GridSearchCV, while it improved around 10 percentage points with the addition of oversampling. From this point, we move to error analysis.

Check variance on best algorithm

In [85]:
```
cv_scores = check_variance(clf_RF, projected_X_train_os, train_labels_os, n_splits=100)
```

In [86]:
```
# Calculate variance.
variance = np.var(cv_scores)

# Plot the histogram.
plt.hist(cv_scores)
plt.title("F1-Score Distribution")
plt.xlabel("F1-Scores")
plt.ylabel("Frequency")

# Calculate the 95 % confidence interval.
print('Variance: {}'.format(variance))
print('The 95 % confidence interval for the f1_score is: ({:.4f}, {:.4f})'
      .format(np.mean(cv_scores) - 1.96*variance**0.5
              ,np.mean(cv_scores) + 1.96*variance**0.5))
```

```
Variance: 1.4857882395481882e-05
The 95 % confidence interval for the f1_score is: (0.6754, 0.6905)
```



Model Error Analysis

With all of the experiments above, we have only been able to improve performance over the out-of-the-box random forest classifier marginally. Let's do some error analysis to understand where the algorithm is making mistakes so that we might have a better chance of future improvement.

For our error analysis, our goal is to look into the confusion matrices that we have been generating to understand further which features existed when certain observations were mislabeled for other classes in our training folds.

In [87]:
```
# Error analysis.
def print_matrix2(matrix, title, features):
    '''This function takes in a matrix, and plots it with the title and
    features to be used as ylabels.
    '''

    fig = plt.figure(figsize = (10, 50))
    ax = fig.add_subplot(111)
    cax = ax.matshow(matrix, cmap = 'Blues')
    plt.title('{}'.format(title))
    #fig.colorbar(cax)
    plt.xticks(np.arange(1, matrix.shape[1] + 1, step=1))
    plt.yticks(np.arange(0, matrix.shape[0], step=1))
    ax.set_yticklabels(features)
    plt.xlabel('Example')
    plt.ylabel('Feature')
    return fig

def find_and_show_errors(fit_labels_true, train_data,
                         pred_labels, indices_test,
                         labels, N=2, J=5):
    '''This function takes in the number of errors, N, the true labels, fit_la
bels_true,
    and the predicted labels, pred_labels.  It outputs the confusion matrix,
    and shows J examples of each of the errors.
    '''

    # Print labels.
    print('Label Legend')
    labels = labels
    for i, label in enumerate(labels):
        print('{}: {}'.format(i, label))

    # Print the confusion matrix.
    print('\nConfusion Matrix')
    cm = confusion_matrix(fit_labels_true, pred_labels)
    plt.show(print_matrix(cm, 'Error Analysis'))

    # Find the highest N confused digits in the CM.
    # These are the highest N numbers in the CM that are not on the diagonal.
    cm_copy = cm.copy()
    np.fill_diagonal(cm_copy, -1)
    error_pairs = np.unravel_index(np.argsort(cm_copy.ravel())[-N:], cm.shape)

    # Create a list of features.
    features_list = train_data_pd.columns.tolist()

    # Loop through the predicted values (error_pairs[0]), one loop for each pr
edicted/true pair.
    print('Top Errors:')
    for [index, value] in enumerate(error_pairs[0]):
        # Store the predicted value and the true value for the error.
        true, pred = error_pairs[0][-index-1], error_pairs[1][-index-1]
        print('\033[1m' + '\n{} is predicted when the true value is {}: {} tim
es.'.format(labels[pred]
```

```
, labels[true]

, cm[true, pred]

)
            )
        print('\033[0m')
        # Find all the examples of the errors.

        miss_arr = np.where((fit_labels_true == labels[true]) & (pred_labels =
= labels[pred]))

        # Print all of the examples of the errors.
        # j is the number of the particular error, row is where it occurs in d
igits and fit_data.
        print('Examples of this error:\n')
        for [j, row] in enumerate(miss_arr[0]):
            if j < J:
                # Print only non-zero features. This saves much space.
                print('Example {}: Non-zero features:'.format(j + 1))
                for i, feature_val in enumerate(train_data[indices_test][row
]):
                    if feature_val != 0:
                        print('\t', features_list[i])
                    else:
                        continue
                print('All other features equal to 0.\n')
            else:
                continue

        # Print plots.
        # Initialize array with number of feature rows and number of mistake c
olumns.
        error_examples = np.empty((len(features_list), J))
        # Iterate through each error.
        for [j, row] in enumerate(miss_arr[0]):
            # j is the number of the particular error, row is where it occurs
 in digits and fit_data.
            if j < J:
                for i, feature_val in enumerate(train_data[indices_test][row
]):
                    # Set the value of the particular feature.
                    error_examples[i, j] = feature_val
            else:
                continue

        # Print error heat maps.
        print_matrix2(error_examples
                      , '{} is predicted when the true value is {}: {} times.'
                      .format(labels[pred], labels[true], cm[true, pred])
                      , features_list)
```

We will check the performance and do error analysis with a single split of the training data into test and training subsets. We do not need to use multiple kfolds because we are trying to evaluate general errors and guide feature engineering. These errors would likely not change much, regardless of the fold, since the variance in predictive performance was low between each fold.

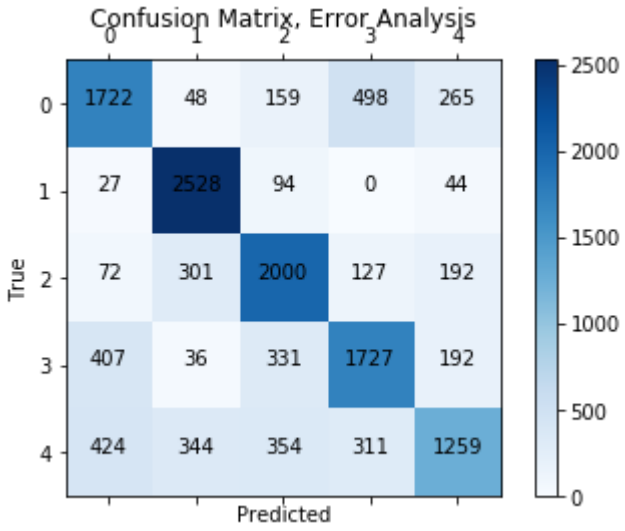
In [88]:
```
# Split training, oversampled, projected data into training and test.
indices = np.asarray(range(len(projected_X_train_os)))
X_train, X_test, y_train, y_test, indices_train, indices_test= train_test_spli
t(projected_X_train_os

, train_labels_os

, indices

, test_size = .25

, stratify = train_labels_os

, random_state = 99)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
print(indices_train.shape)
print(indices_test.shape)
```

```
(40383, 30)
(13462, 30)
(40383,)
(13462,)
(40383,)
(13462,)
```

In [89]:
```
# Fit the optimal model.
clf_RF = RandomForestClassifier(max_depth=80, min_samples_split=6, criterion=
'entropy')
clf_RF.fit(X_train, y_train)
pred_labels = clf_RF.predict(X_test)

labels  = ['Adoption', 'Died', 'Euthanasia', 'Return_to_owner', 'Transfer']
# Find the errors for the optimal model.
find_and_show_errors(fit_labels_true=y_test, train_data=train_data_os
                     , pred_labels=pred_labels, indices_test=indices_test
                     , labels=labels, N=10, J=40
                    )
```

Label Legend
0: Adoption
1: Died
2: Euthanasia
3: Return_to_owner
4: Transfer

Confusion Matrix



Top Errors:

Return_to_owner is predicted when the true value is Adoption: 498 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Dachshund
White
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Great Pyrenees
White
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Labrador Retriever
Yellow
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Labrador Retriever
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Chihuahua Shorthair
Tan
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
State(Fixed=1)
Dachshund
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Manx
Black
AgeuponOutcome_norm

```
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Australian Kelpie
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         Yellow
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Parson Russell Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shepherd
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Affenpinscher
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
```

```
         Gender(Male=1)
         Plott Hound
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Maltese
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Miniature Poodle
         Buff
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         American Pit Bull Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Cairn Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         German Shepherd
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Skye Terrier
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         German Shepherd
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Australian Cattle Dog
         Red
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Buff
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Jack Russell Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Boston Terrier
         White
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Border Collie
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
         Chihuahua Longhair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
         State(Fixed=1)
         Harrier
         Tricolor
```

AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Cairn Terrier
White
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
German Shepherd
Black
AgeuponOutcome_norm
All other features equal to 0.

Adoption is predicted when the true value is Transfer: 424 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
German Shepherd
Tan
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Chihuahua Longhair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Pit Bull
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Domestic Shorthair
Brown Tabby

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Border Collie
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Norwich Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Chesa Bay Retr
         Chocolate
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Domestic Medium Hair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shepherd
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
```

```
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Miniature Schnauzer
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Soft Coated Wheaten Terrier
         Buff
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Longhair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Medium Hair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Torbie
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Staffordshire
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         German Shepherd
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Boxer
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Kelpie
```

```
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Schnauzer Giant
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         Buff
         AgeuponOutcome_norm
All other features equal to 0.
```

Example 36: Non-zero features:
Domestic Shorthair
Blue Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
State(Fixed=1)
Great Pyrenees
White
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
Gender(Male=1)
Chihuahua Shorthair
Tan
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Maltese
White
AgeuponOutcome_norm
All other features equal to 0.

Adoption is predicted when the true value is Return_to_owner: 407 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Russian Blue
Blue
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Labrador Retriever
Black
AgeuponOutcome_norm
All other features equal to 0.

```
Example 3: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Cardigan Welsh Corgi
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Collie Smooth
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Jack Russell Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Pit Bull
```

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Yorkshire Terrier
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Anatol Shepherd
         Sable
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         German Shepherd
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Cardigan Welsh Corgi
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 18: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Dachshund
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shorthair Pointer
         Chocolate
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Miniature Poodle
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         Beagle
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shepherd
         Sable
```

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         Gender(Male=1)
         Cairn Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pug
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Longhair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Staffordshire
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
```

```
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         French Bulldog
         Fawn
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Shih Tzu
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Medium Hair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
         State(Fixed=1)
         Siamese
         Lilac Point
         AgeuponOutcome_norm
```

All other features equal to 0.

Euthanasia is predicted when the true value is Transfer: 354 times.

Examples of this error:

Example 1: Non-zero features:
Labrador Retriever
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
Dachshund Wirehair
Tan
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
Gender(Male=1)
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
State(Fixed=1)
Domestic Shorthair
White

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Domestic Longhair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         Labrador Retriever
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Mastiff
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         Domestic Shorthair
         Torbie
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Gender(Male=1)
         Domestic Medium Hair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         Gender(Male=1)
```

```
         Queensland Heeler
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         Gender(Male=1)
         German Shepherd
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chinese Sharpei
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         Snowshoe
         Seal Point
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
```

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         Gender(Male=1)
         Bichon Frise
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Shih Tzu
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         Miniature Poodle
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

Example 34: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Pit Bull
Brown Brindle
AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
Gender(Male=1)
Siamese
Seal Point
AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
Gender(Male=1)
Pit Bull
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
Chihuahua Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
Gender(Male=1)
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Medium Hair
Black
AgeuponOutcome_norm
All other features equal to 0.

Died is predicted when the true value is Transfer: 344 times.

Examples of this error:

Example 1: Non-zero features:

```
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Cream Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
         Domestic Medium Hair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         Gender(Male=1)
         Domestic Medium Hair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         Manx
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 10: Non-zero features:
         Domestic Shorthair
         Calico
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Black
All other features equal to 0.

Example 12: Non-zero features:
         Domestic Shorthair
         Torbie
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         Gender(Male=1)
         Labrador Retriever
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         Domestic Shorthair
         White
All other features equal to 0.

Example 18: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 19: Non-zero features:
         Domestic Shorthair
         Black Smoke
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         Gender(Male=1)
         Domestic Medium Hair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         Domestic Shorthair
         White
All other features equal to 0.

Example 23: Non-zero features:
         Gender(Male=1)
         German Shepherd
         White
All other features equal to 0.

Example 24: Non-zero features:
         Gender(Male=1)
         Domestic Medium Hair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         Domestic Shorthair
         White
All other features equal to 0.

Example 26: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         Domestic Shorthair
         Black
All other features equal to 0.

Example 28: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
```

```
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         Gender(Male=1)
         Siamese
         Flame Point
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         Domestic Shorthair
         Brown Tabby
All other features equal to 0.

Example 36: Non-zero features:
         Gender(Male=1)
         Pit Bull
         White
All other features equal to 0.
```

Example 37: Non-zero features:
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
Domestic Shorthair
Black
All other features equal to 0.

Euthanasia is predicted when the true value is Return_to_owner: 331 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Border Collie
White
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Pit Bull
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
Gender(Male=1)
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

```
Example 5: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Great Dane
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Gender(Male=1)
         German Shepherd
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
```

```
        Gender(Male=1)
        Pit Bull
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
        Gender(Male=1)
        Boxer
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
        State(Fixed=1)
        Shih Tzu
        Gray
        AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
        State(Fixed=1)
        German Shepherd
        Black
        AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
        State(Fixed=1)
        Pit Bull
        Tan
        AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
        Pit Bull
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
        Gender(Male=1)
        Pit Bull
        Blue
        AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
        Gender(Male=1)
        Pit Bull
        Red
        AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
        State(Fixed=1)
        Pit Bull
```

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         Pit Bull
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         Blue Merle
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Calico
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chesa Bay Retr
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Australian Cattle Dog
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         Gender(Male=1)
         Basset Hound
```

Tricolor
AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
Gender(Male=1)
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
State(Fixed=1)
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Return_to_owner is predicted when the true value is Transfer: 311 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Labrador Retriever
Tan
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Boxer
White
AgeuponOutcome_norm

```
All other features equal to 0.

Example 3: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Beauceron
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Border Collie
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
         Labrador Retriever
         Yellow
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Siberian Husky
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Boston Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Dachshund
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Boxer
         White
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Pug
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Boston Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         German Shepherd
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Siberian Husky
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Domestic Longhair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Snowshoe
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         Gender(Male=1)
         Chinese Sharpei
         White
```

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Great Pyrenees
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Miniature Schnauzer
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         Gender(Male=1)
         Shiba Inu
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Great Dane
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         Boxer
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
```

```
         Pit Bull
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Dachshund
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Great Pyrenees
         Cream
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         Pink
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shepherd
         Tan
         AgeuponOutcome_norm
All other features equal to 0.
```

Example 34: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Pointer
White
AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
State(Fixed=1)
Pit Bull
Red
AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
State(Fixed=1)
Domestic Shorthair
Calico
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Shih Tzu
White
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Italian Greyhound
Gray
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Catahoula
Blue Merle
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Beagle
Tricolor
AgeuponOutcome_norm
All other features equal to 0.

Died is predicted when the true value is Euthanasia: 301 times.

Examples of this error:

Example 1: Non-zero features:
Domestic Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
Gender(Male=1)
Chihuahua Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
Gender(Male=1)
Chihuahua Shorthair
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
Domestic Medium Hair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
Domestic Shorthair
Torbie
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
Gender(Male=1)

```
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Gender(Male=1)
         Chihuahua Shorthair
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 18: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         Domestic Shorthair
```

        Calico
        AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
        Gender(Male=1)
        Domestic Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
        Domestic Shorthair
        Black
        AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
        Chihuahua Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
        Domestic Shorthair
        Brown Tabby
        AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
        Gender(Male=1)
        Domestic Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
        Gender(Male=1)
        Domestic Shorthair
        Brown Tabby
        AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
        Gender(Male=1)
        Domestic Medium Hair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
        Domestic Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:

Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
Domestic Shorthair
Calico
AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Transfer is predicted when the true value is Adoption: 265 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Domestic Medium Hair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Neapolitan Mastiff
White
AgeuponOutcome_norm
All other features equal to 0.

```
Example 3: Non-zero features:
         State(Fixed=1)
         Miniature Schnauzer
         Blue Merle
         AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Border Collie
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         English Springer Spaniel
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Red
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
```

```
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Domestic Medium Hair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Gender(Male=1)
         Siamese
         Lilac Point
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         Buff
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Gender(Male=1)
         Siamese
         Seal Point
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 18: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Dachshund
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Cairn Terrier
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Miniature Poodle
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Dachshund
         Red
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 25: Non-zero features:
         State(Fixed=1)
         Treeing Walker Coonhound
         Blue Tick
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Longhair
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Dachshund
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Cairn Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Australian Cattle Dog
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Boxer
         White
         AgeuponOutcome_norm
```

```
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Great Dane
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
         Carolina Dog
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
         State(Fixed=1)
         Rat Terrier
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Orange Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
```

White
AgeuponOutcome_norm
All other features equal to 0.

Transfer is predicted when the true value is Euthanasia: 192 times.

Examples of this error:

Example 1: Non-zero features:
Domestic Shorthair
Calico
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
Gender(Male=1)
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
Domestic Shorthair
Blue Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
State(Fixed=1)
Rottweiler
Tan
AgeuponOutcome_norm

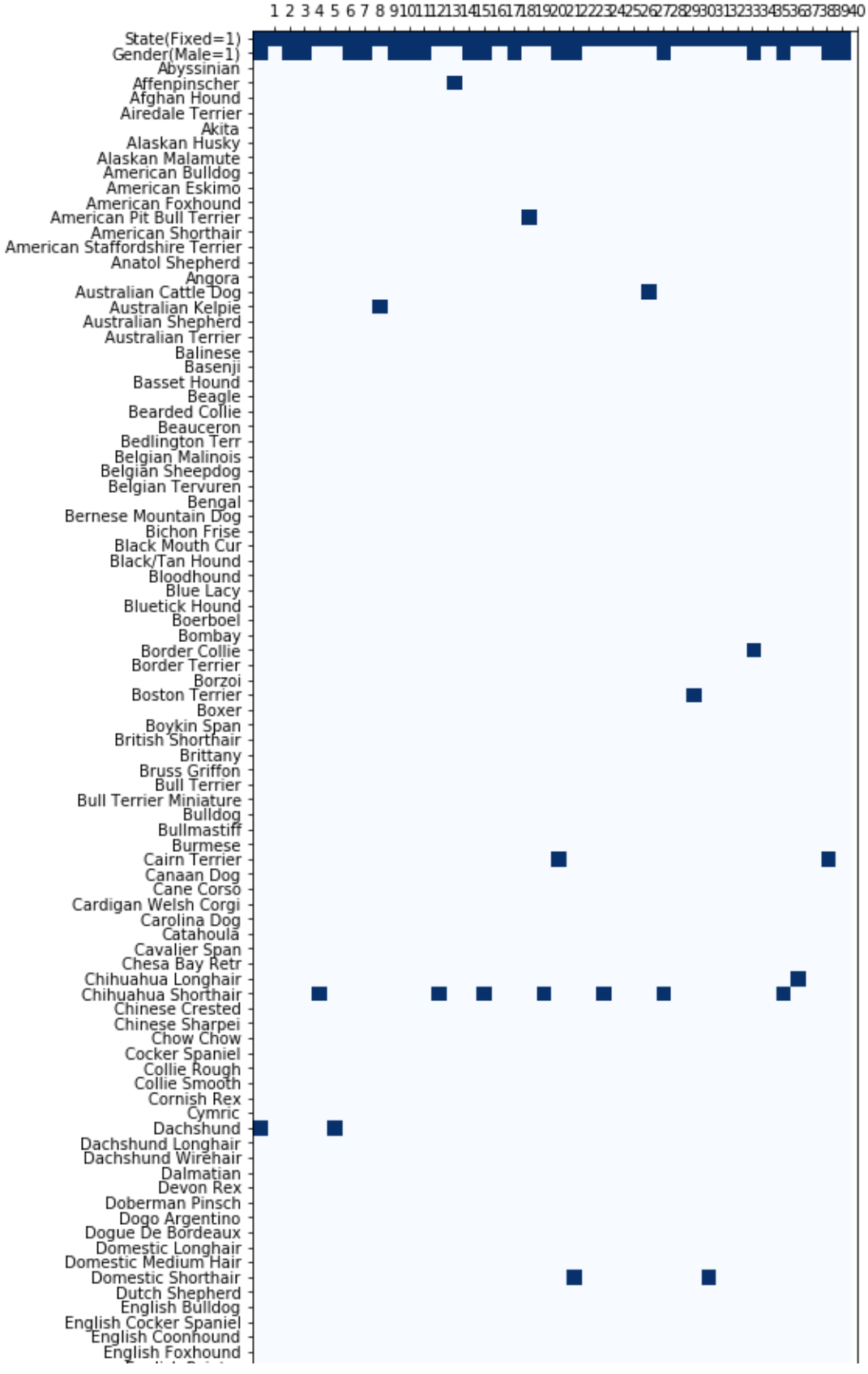
```
All other features equal to 0.

Example 9: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
```

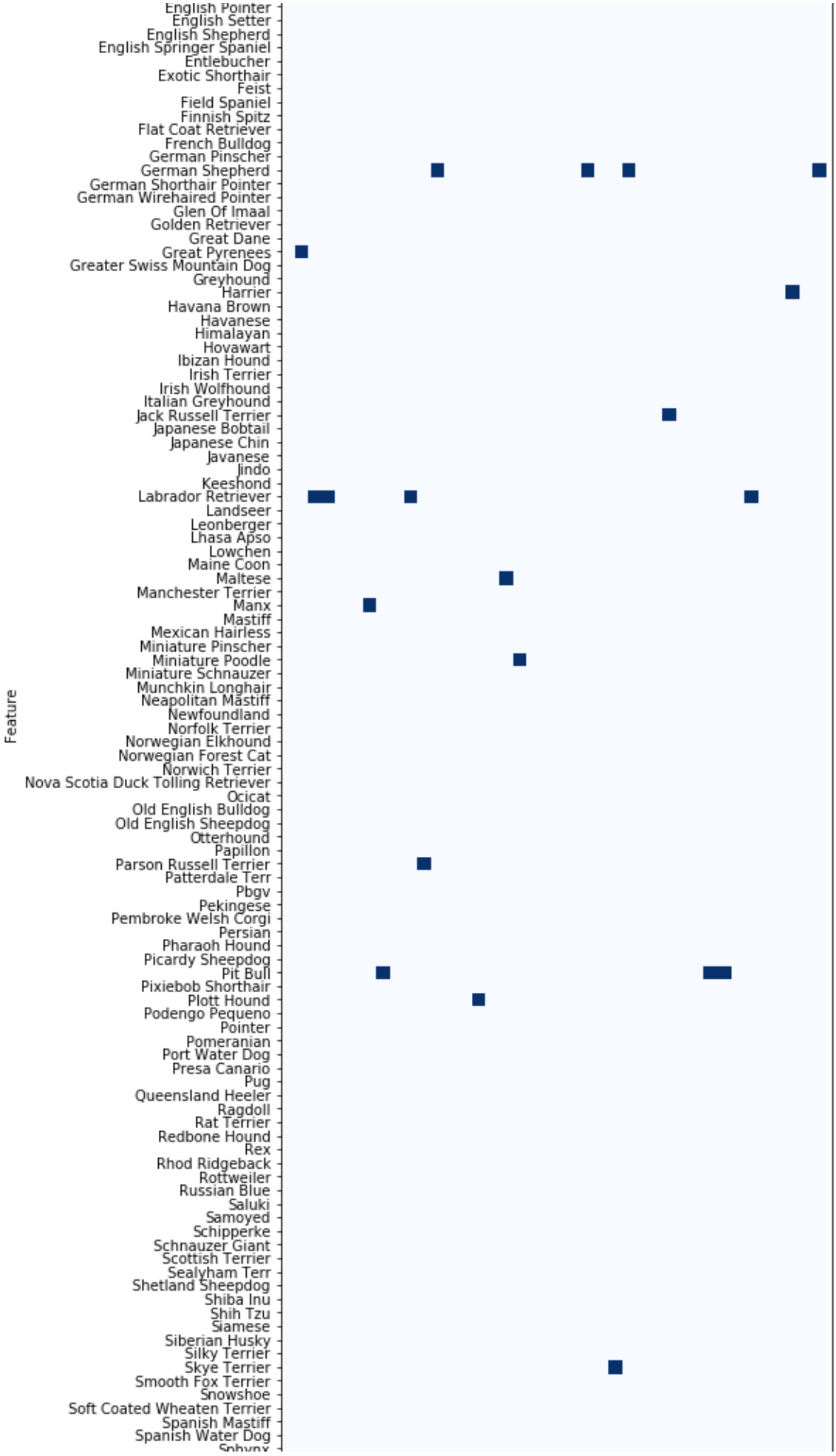
```
All other features equal to 0.

Example 18: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Rottweiler
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         Domestic Shorthair
         Calico
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         Domestic Shorthair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         Domestic Shorthair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         Pit Bull
```

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         Domestic Shorthair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         Domestic Shorthair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
```

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         Domestic Shorthair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
         Domestic Shorthair
         Calico
         AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

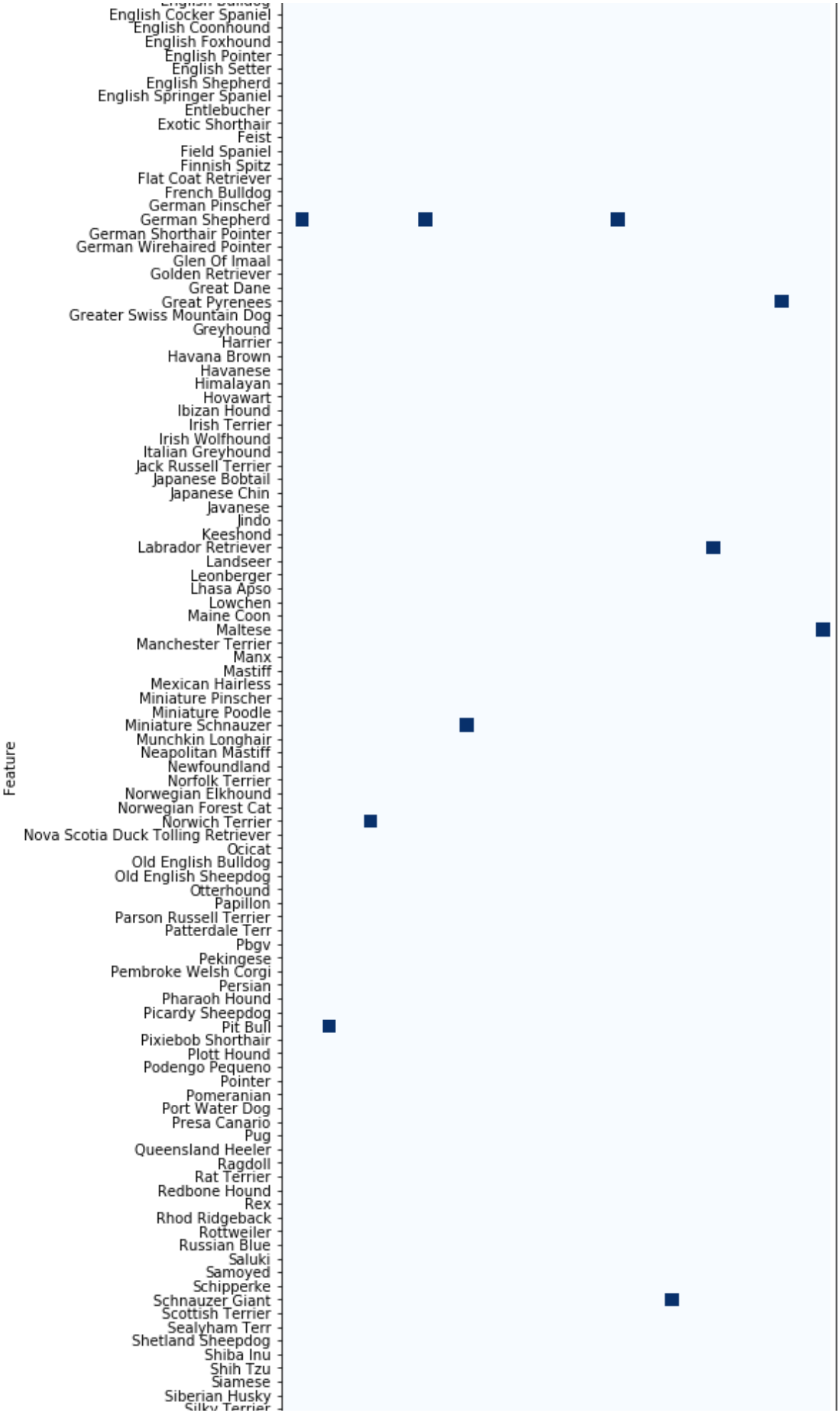
Return_to_owner is predicted when the true value is Adoption: 498 times.




Feature
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx

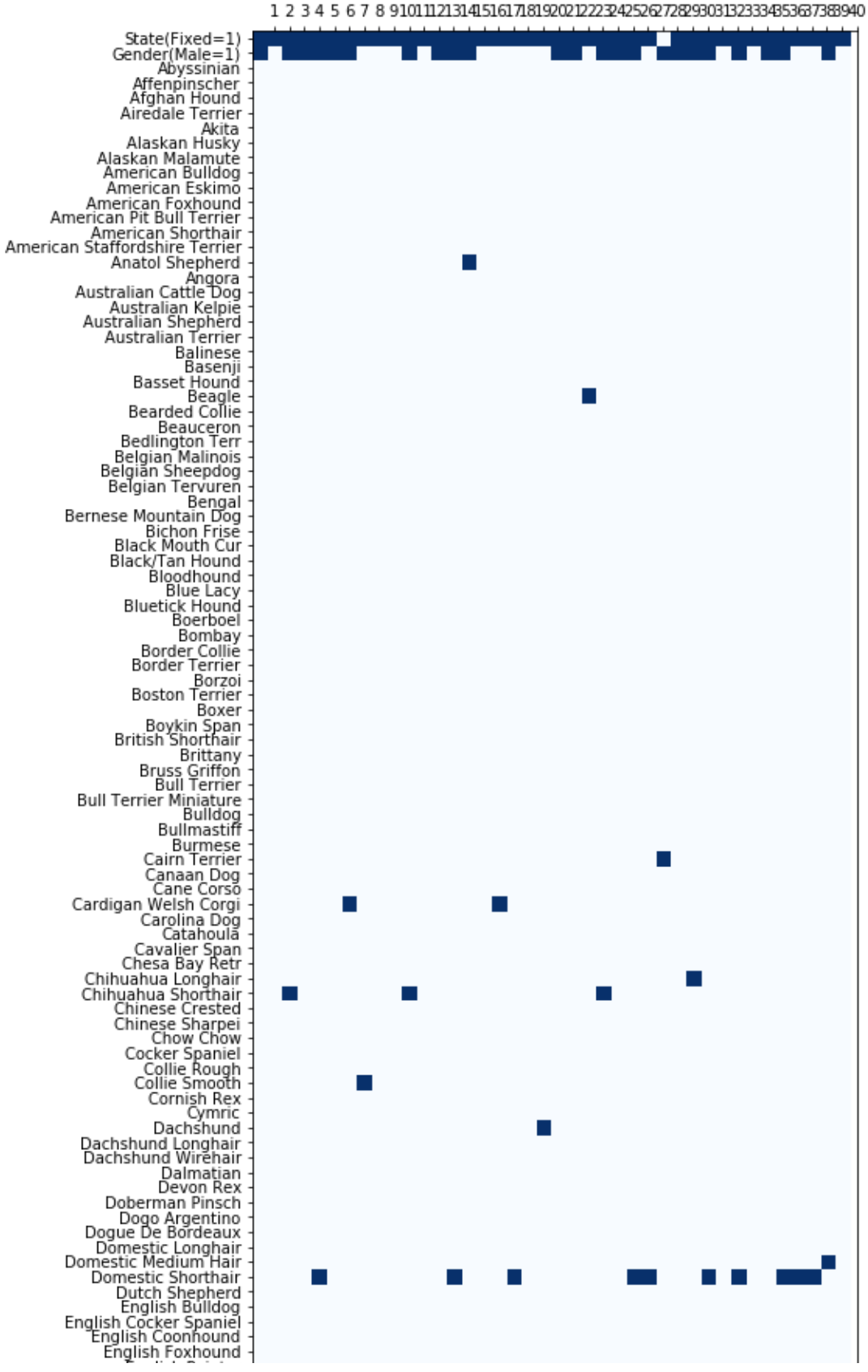

Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

Adoption is predicted when the true value is Transfer: 424 times.


English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

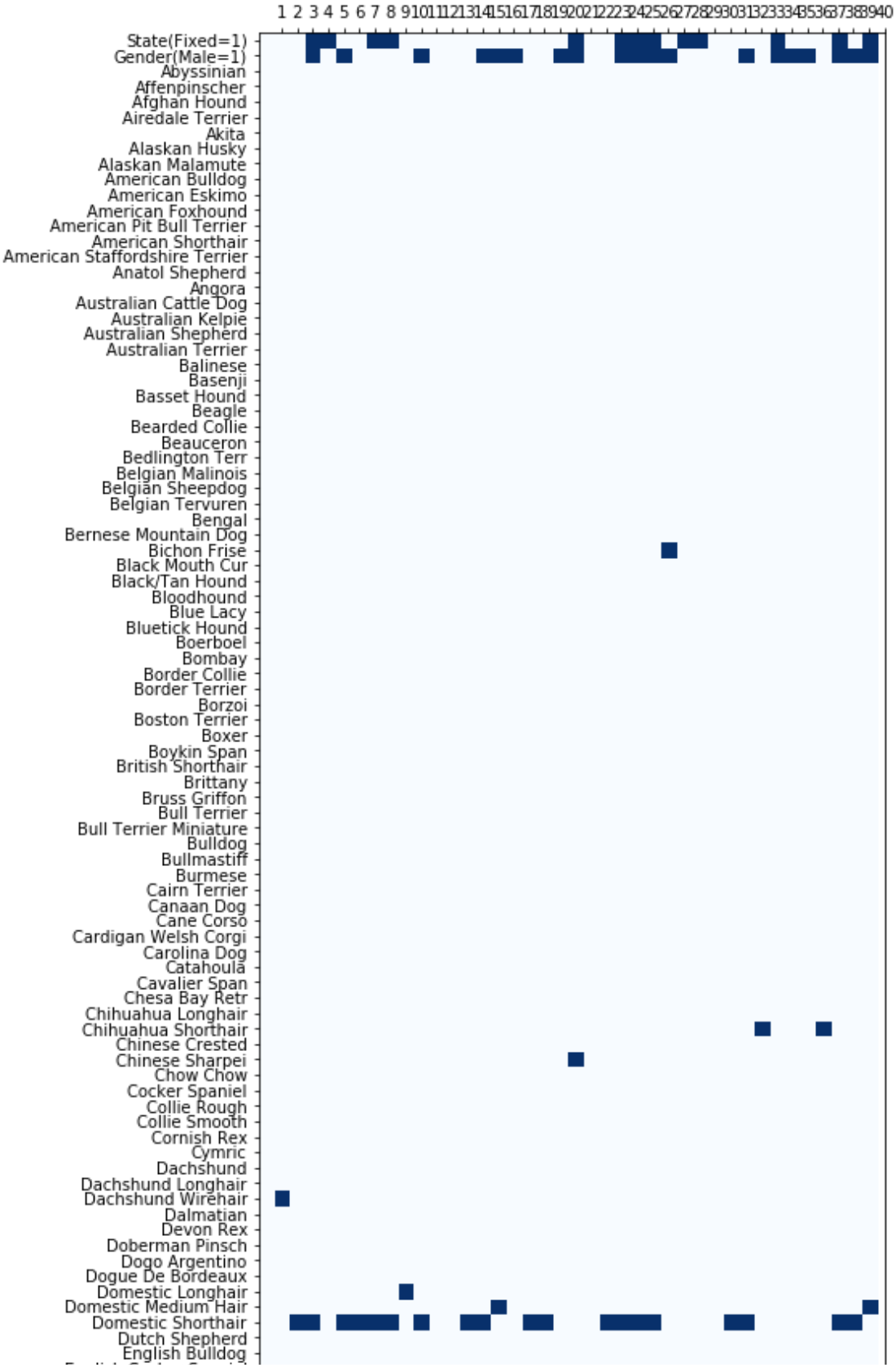
Adoption is predicted when the true value is Return_to_owner: 407 times.

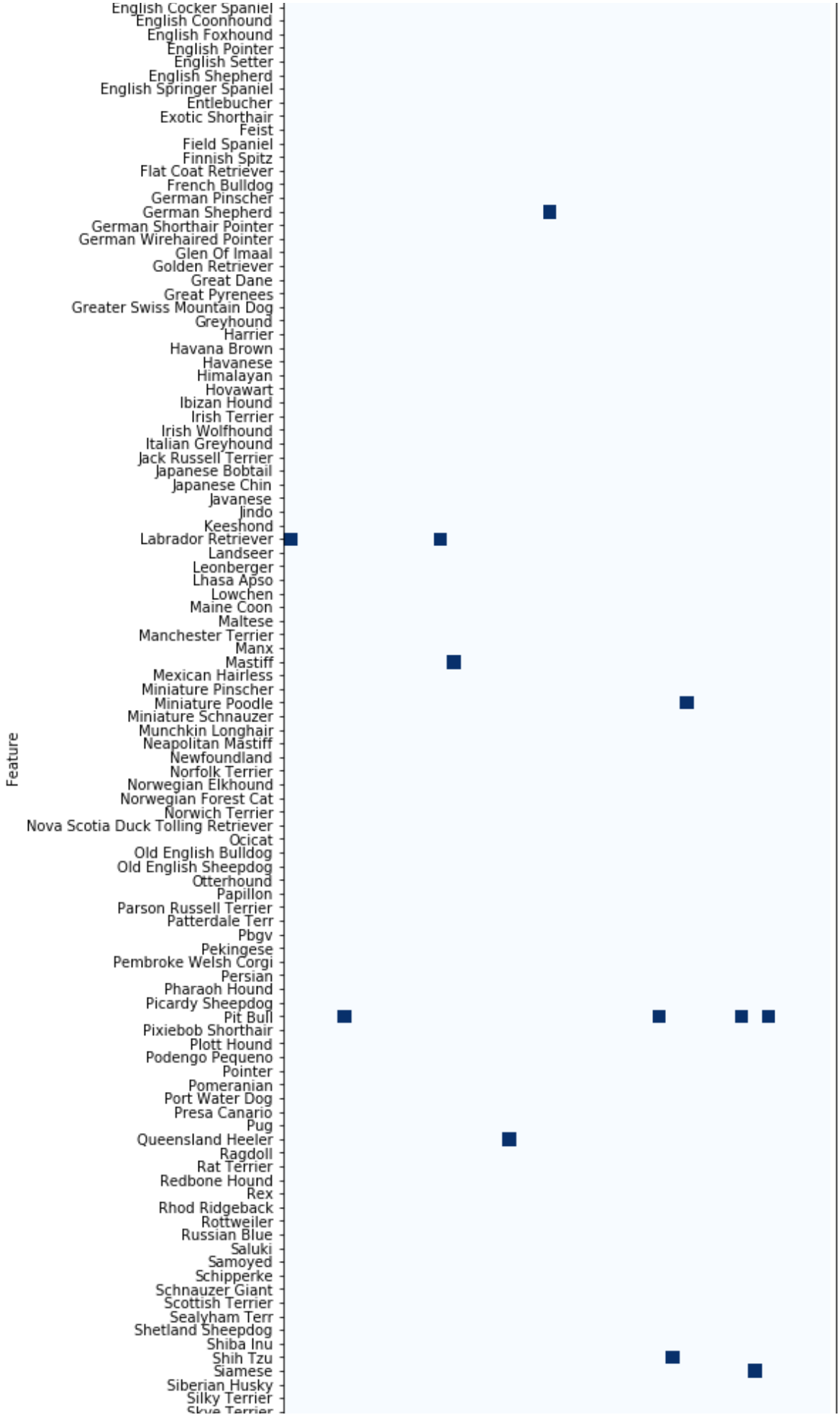



Feature
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx


Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

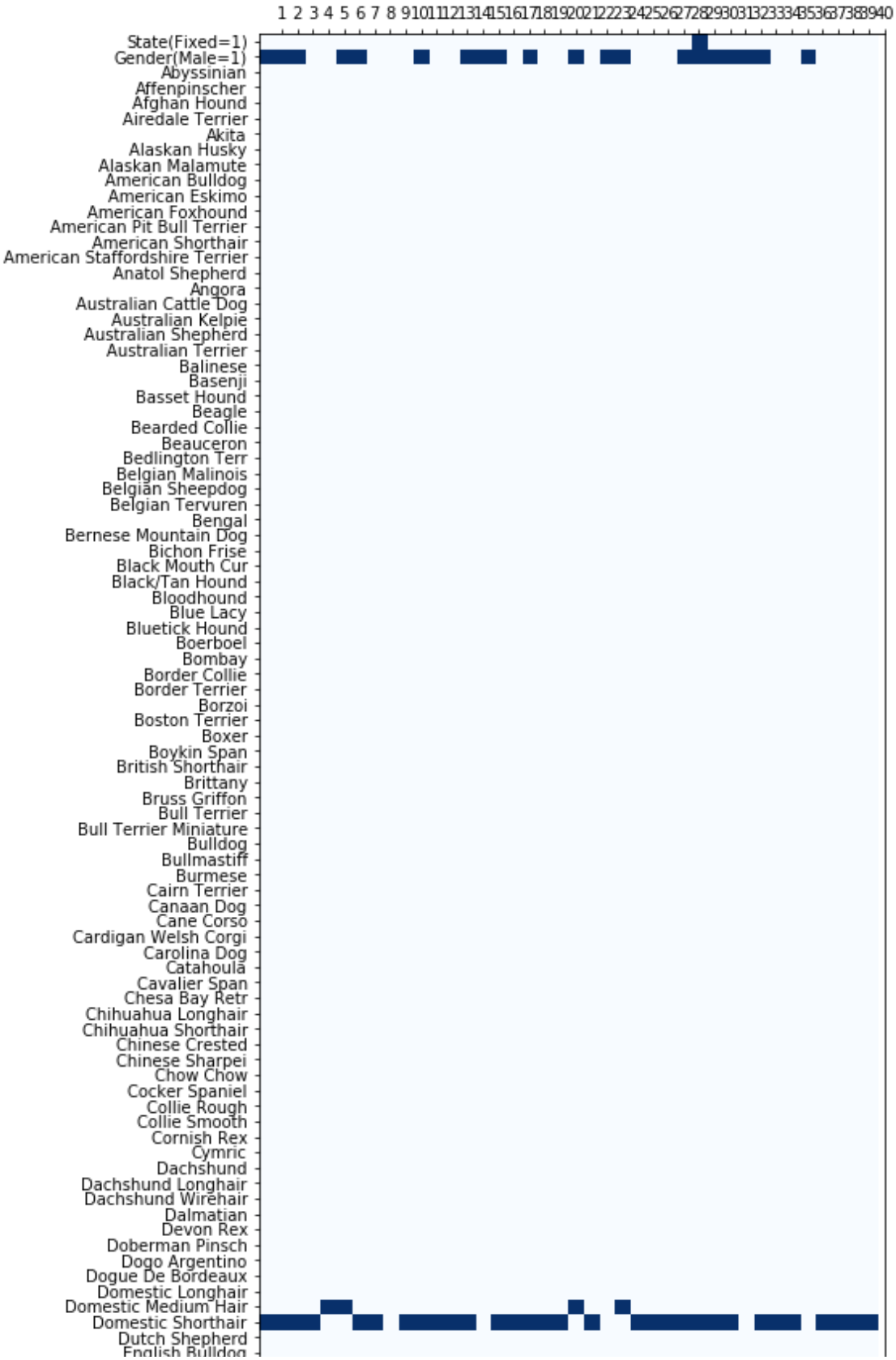
Euthanasia is predicted when the true value is Transfer: 354 times.

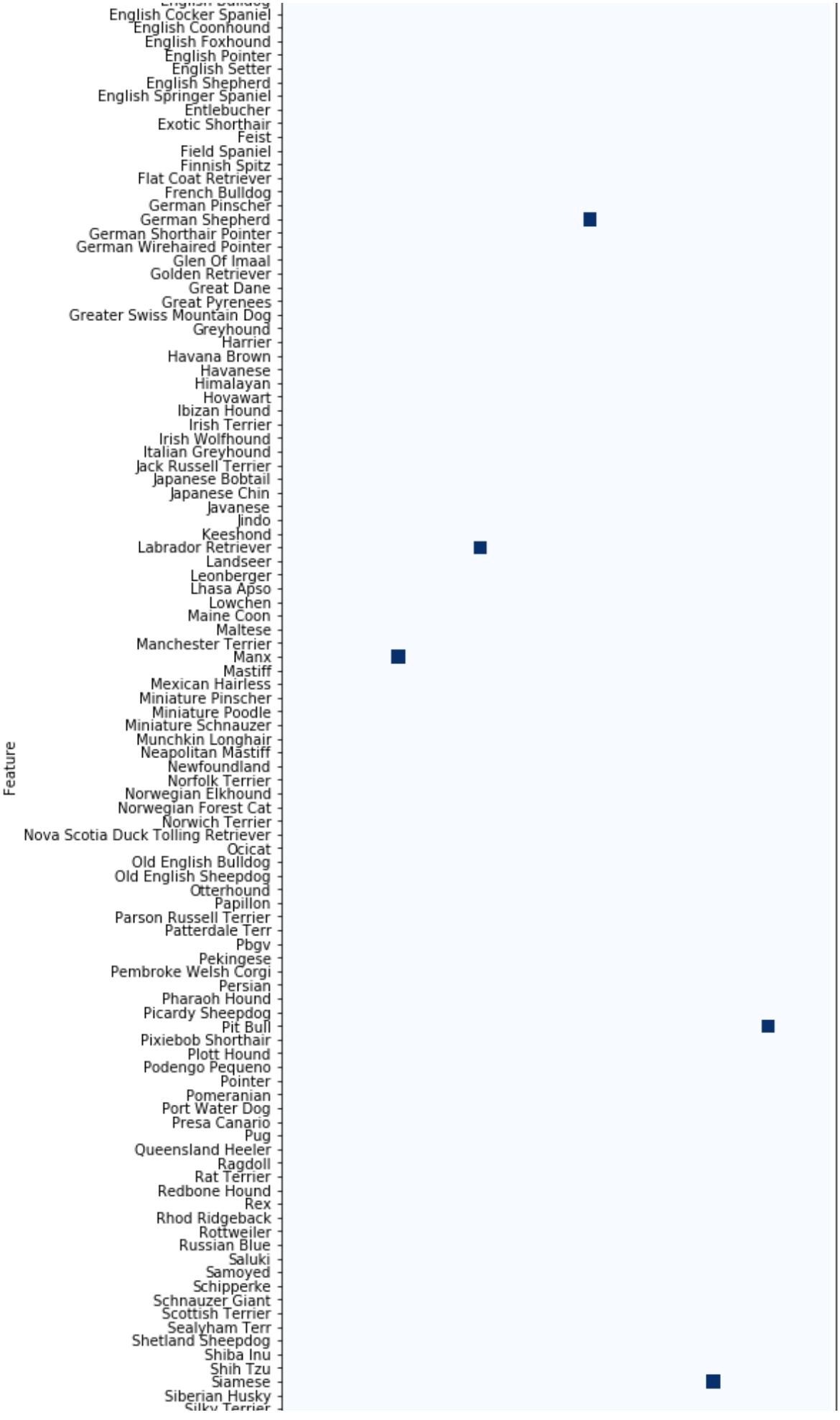


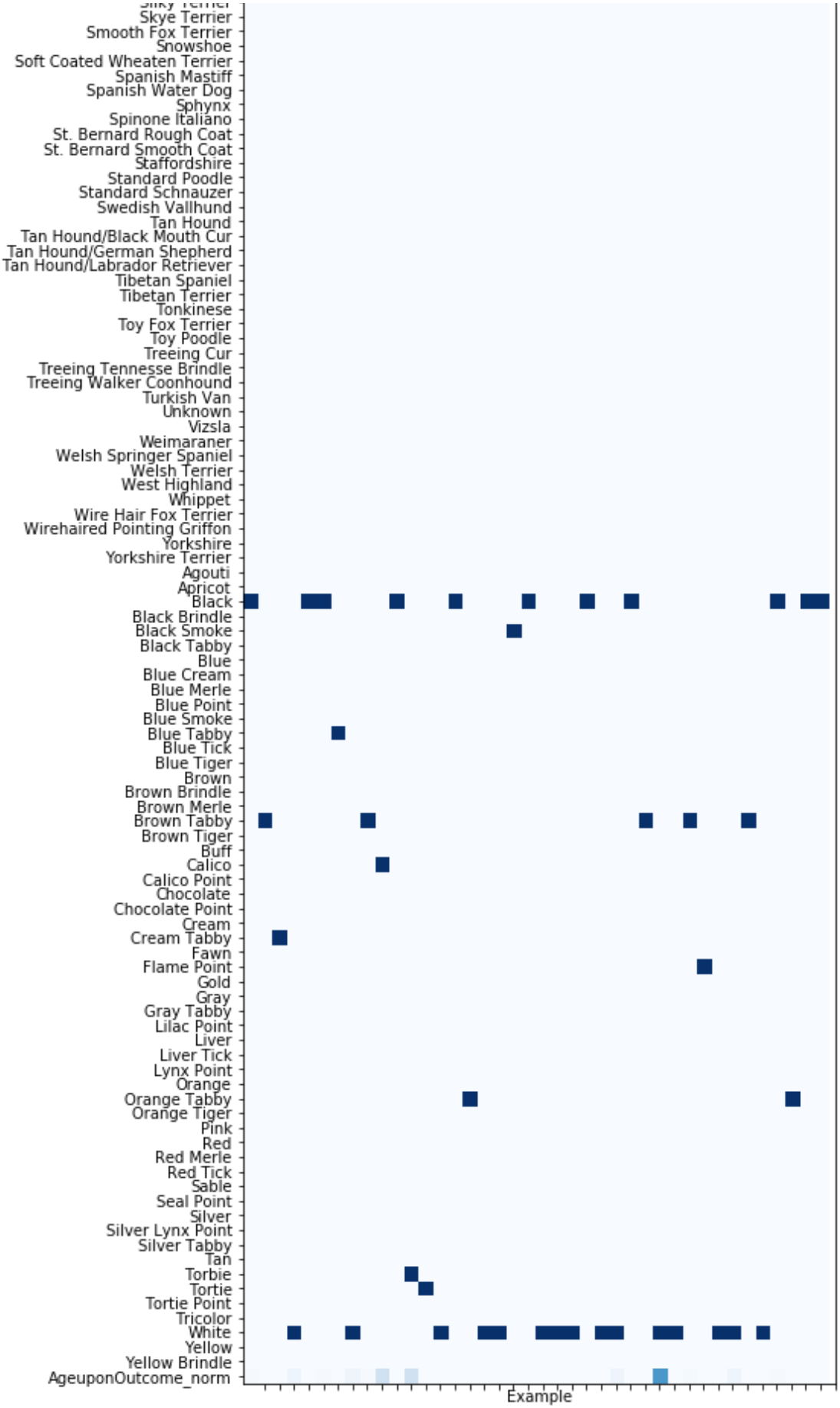

Feature
English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

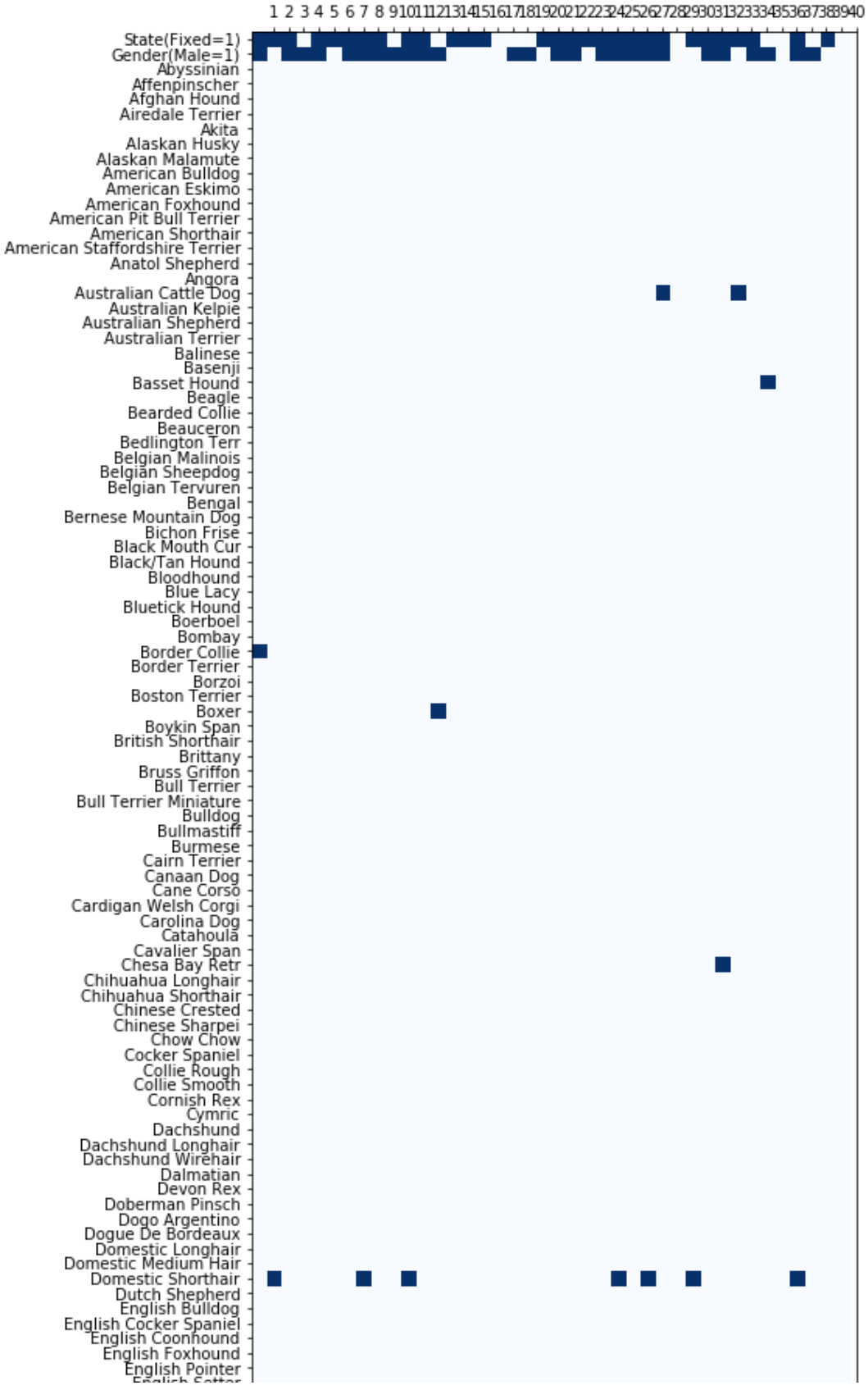
Died is predicted when the true value is Transfer: 344 times.



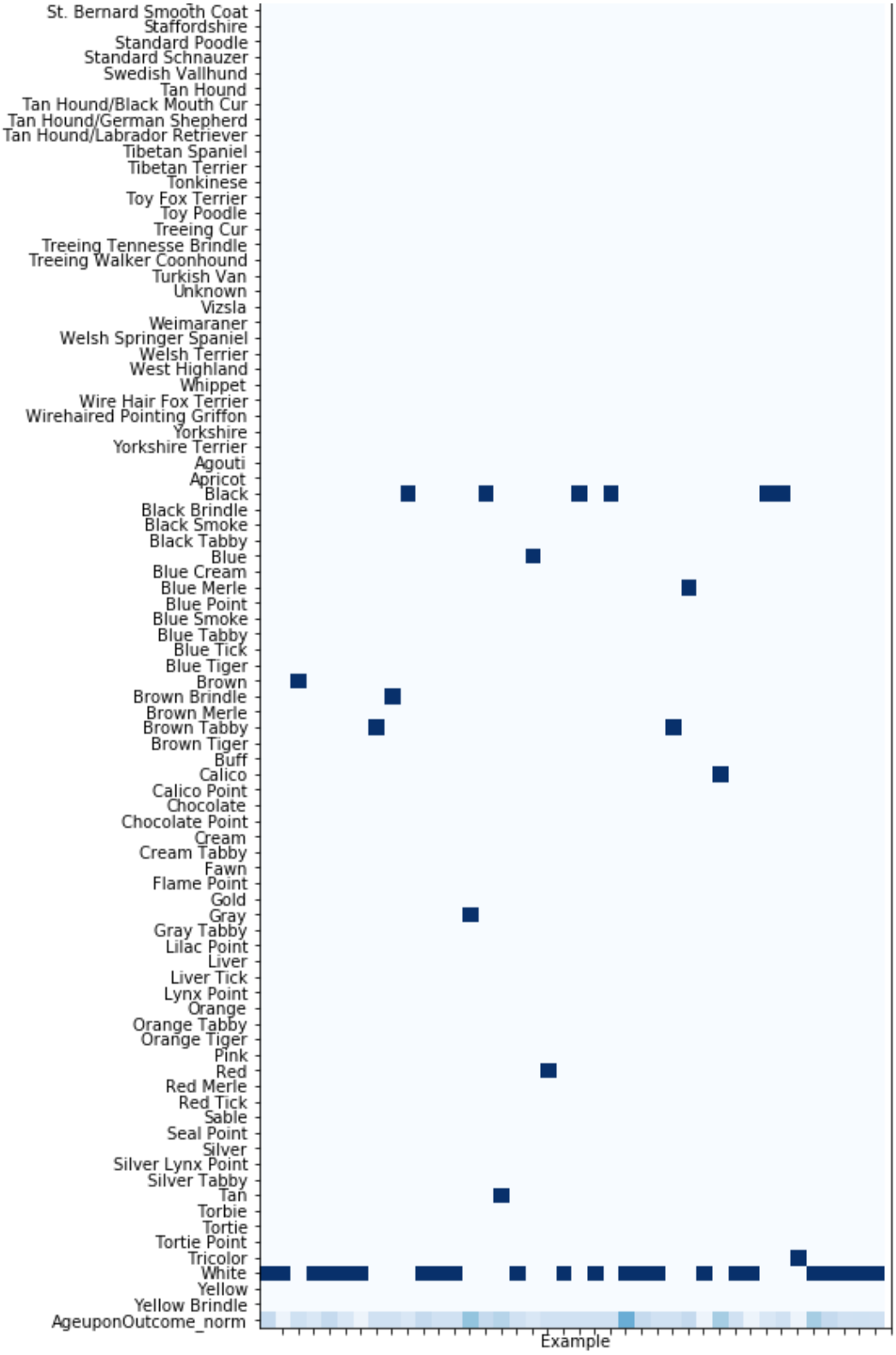

Feature
English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

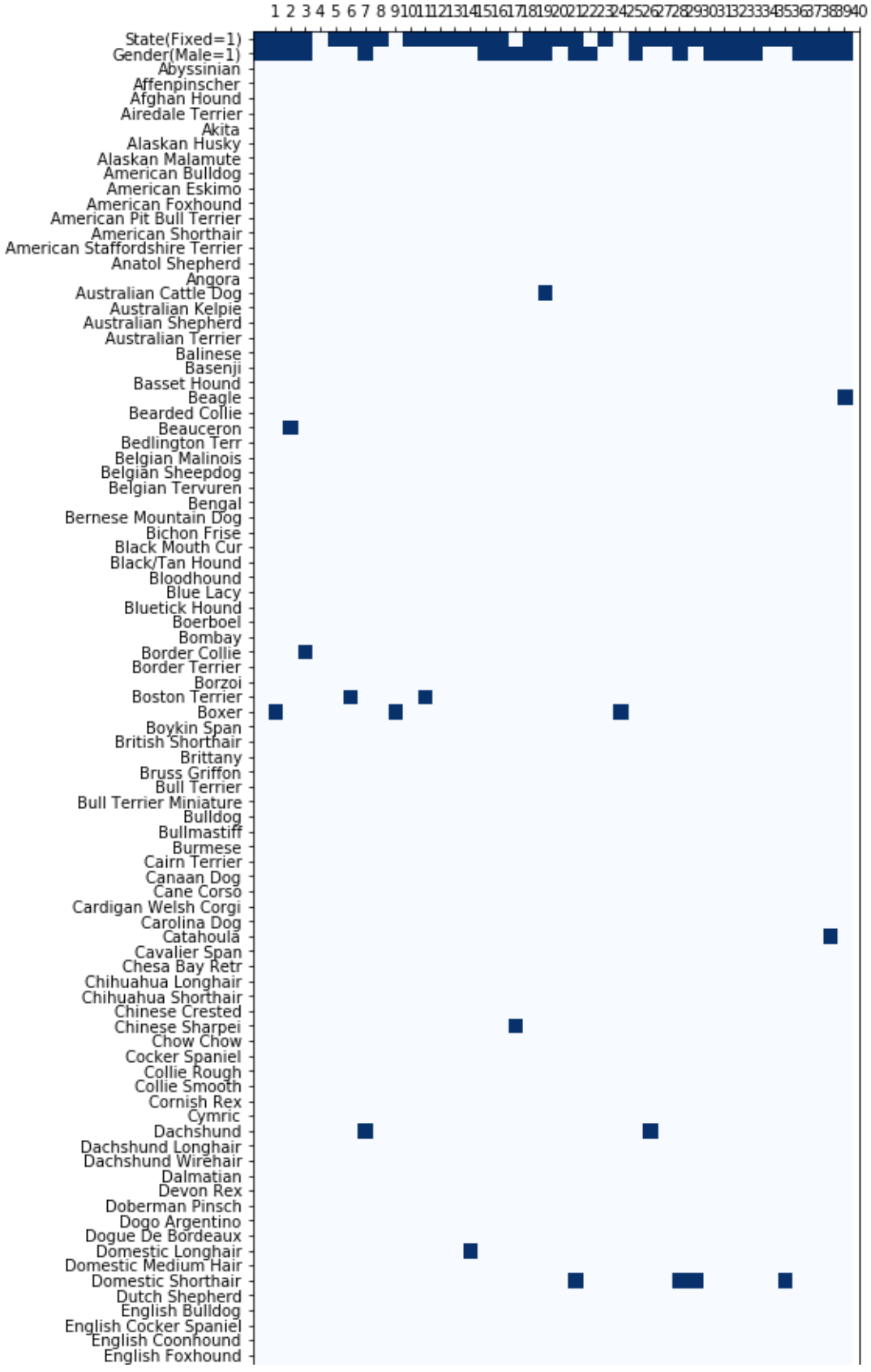
Euthanasia is predicted when the true value is Return_to_owner: 331 times.

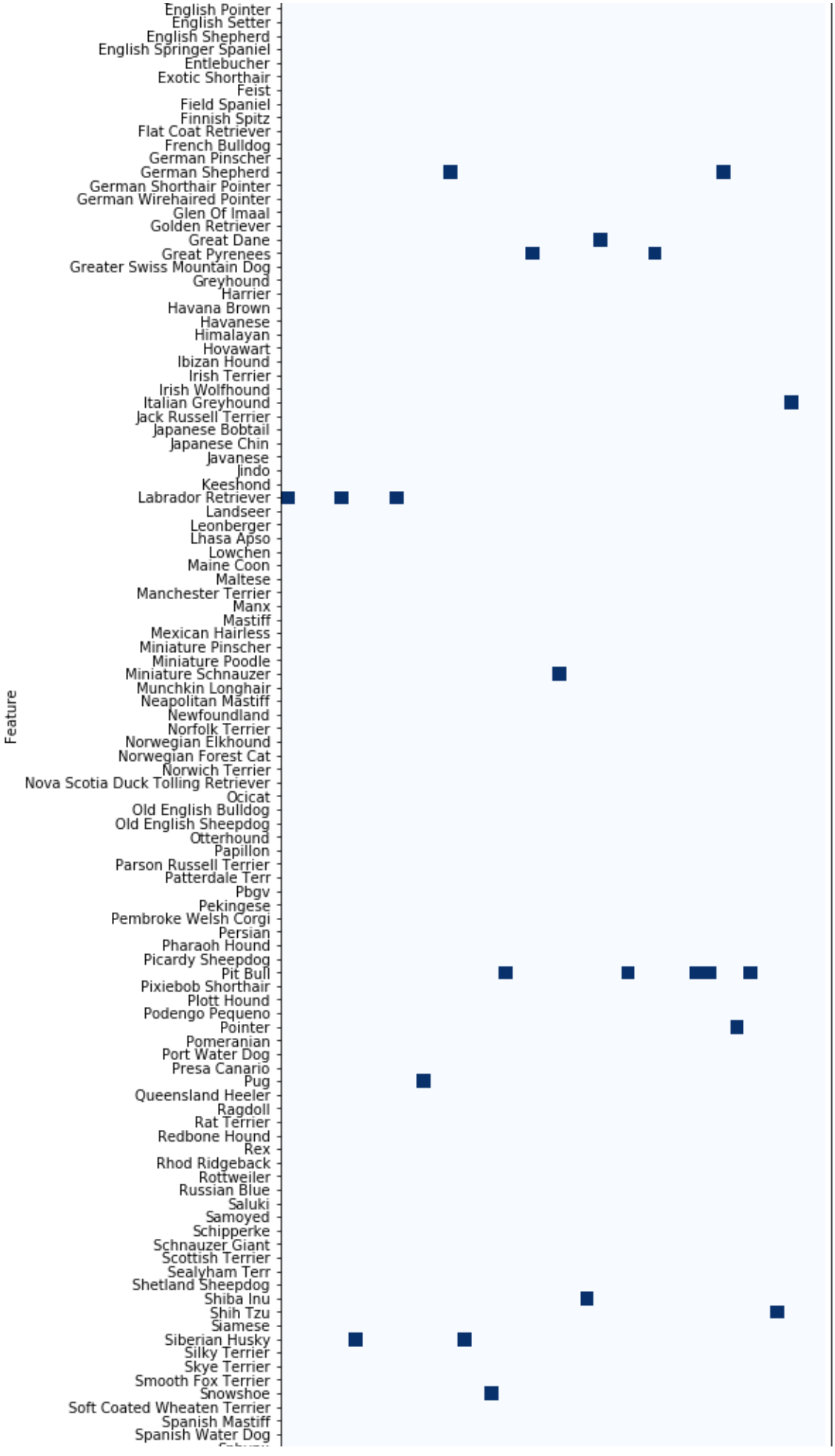



English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
Feature


St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

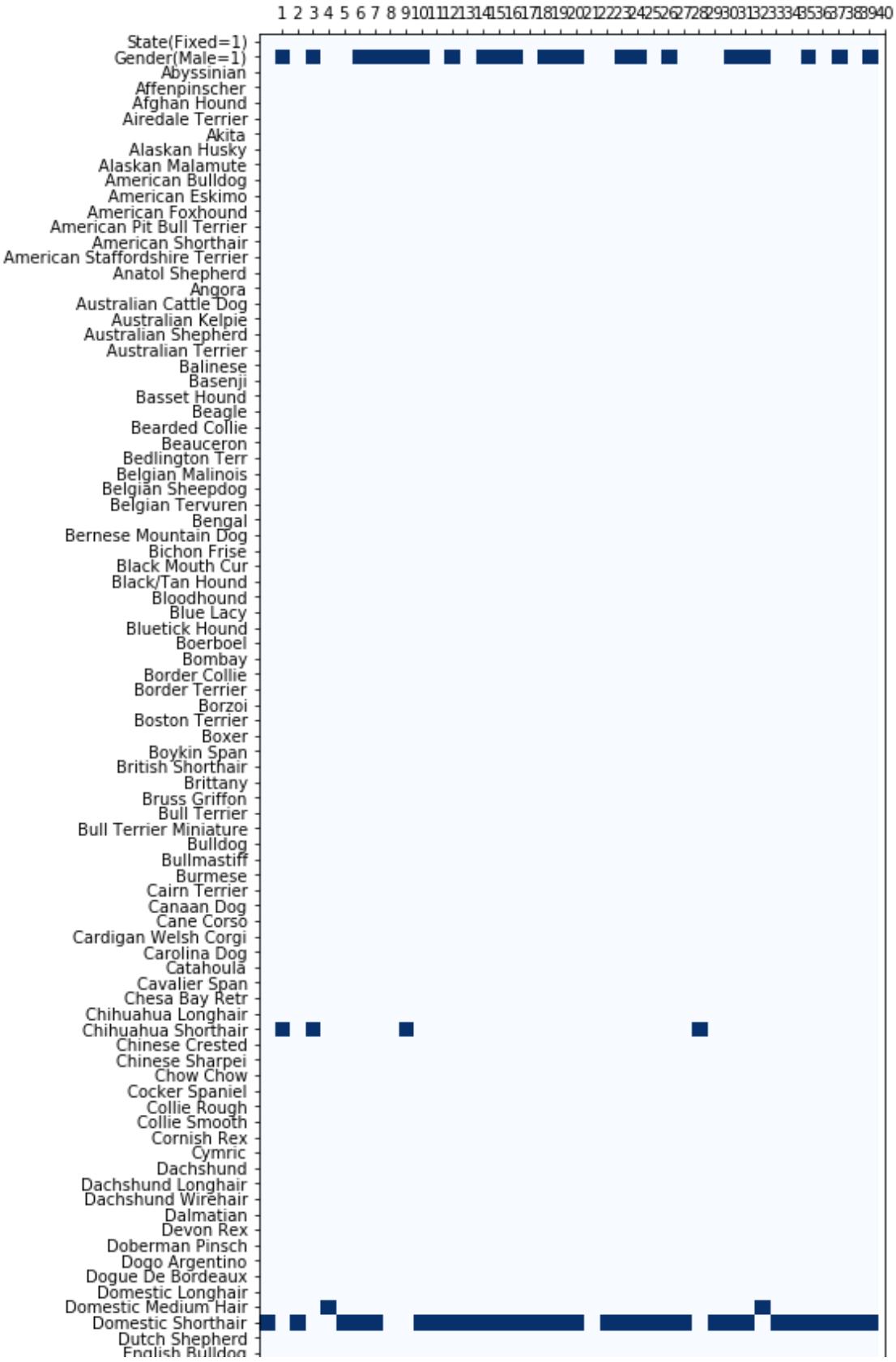
Return_to_owner is predicted when the true value is Transfer: 311 times.

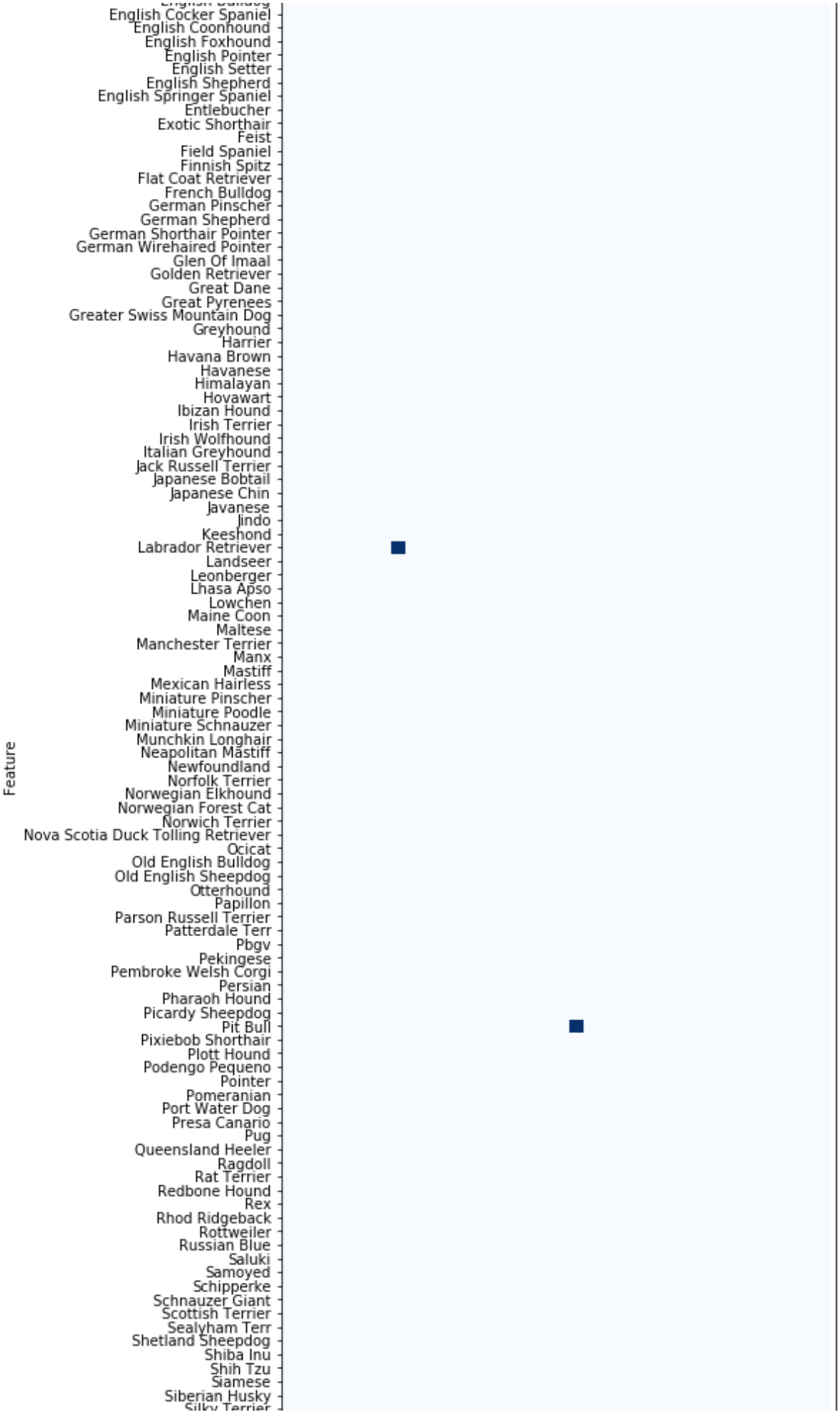


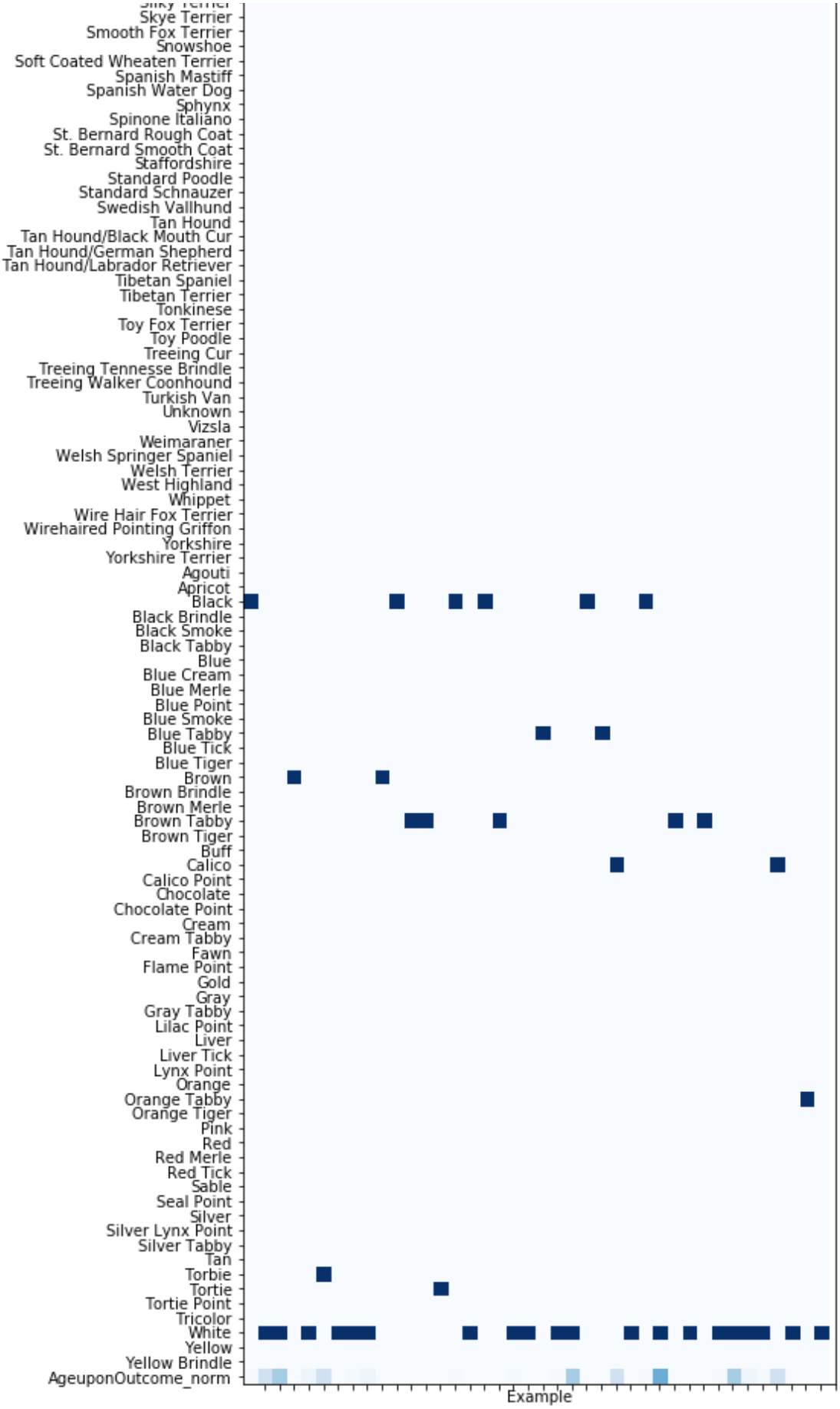

Feature
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog


Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

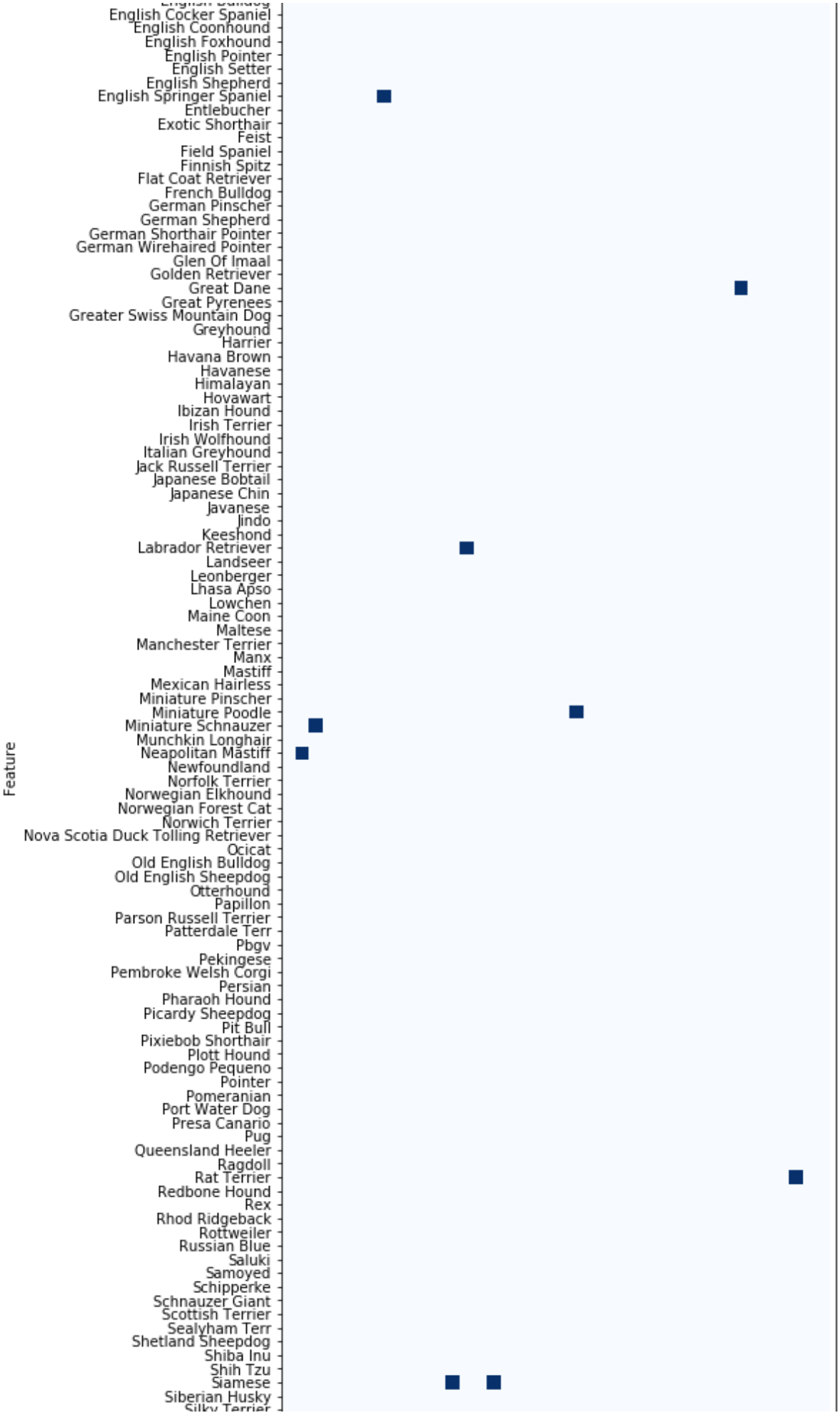
Died is predicted when the true value is Euthanasia: 301 times.




English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature

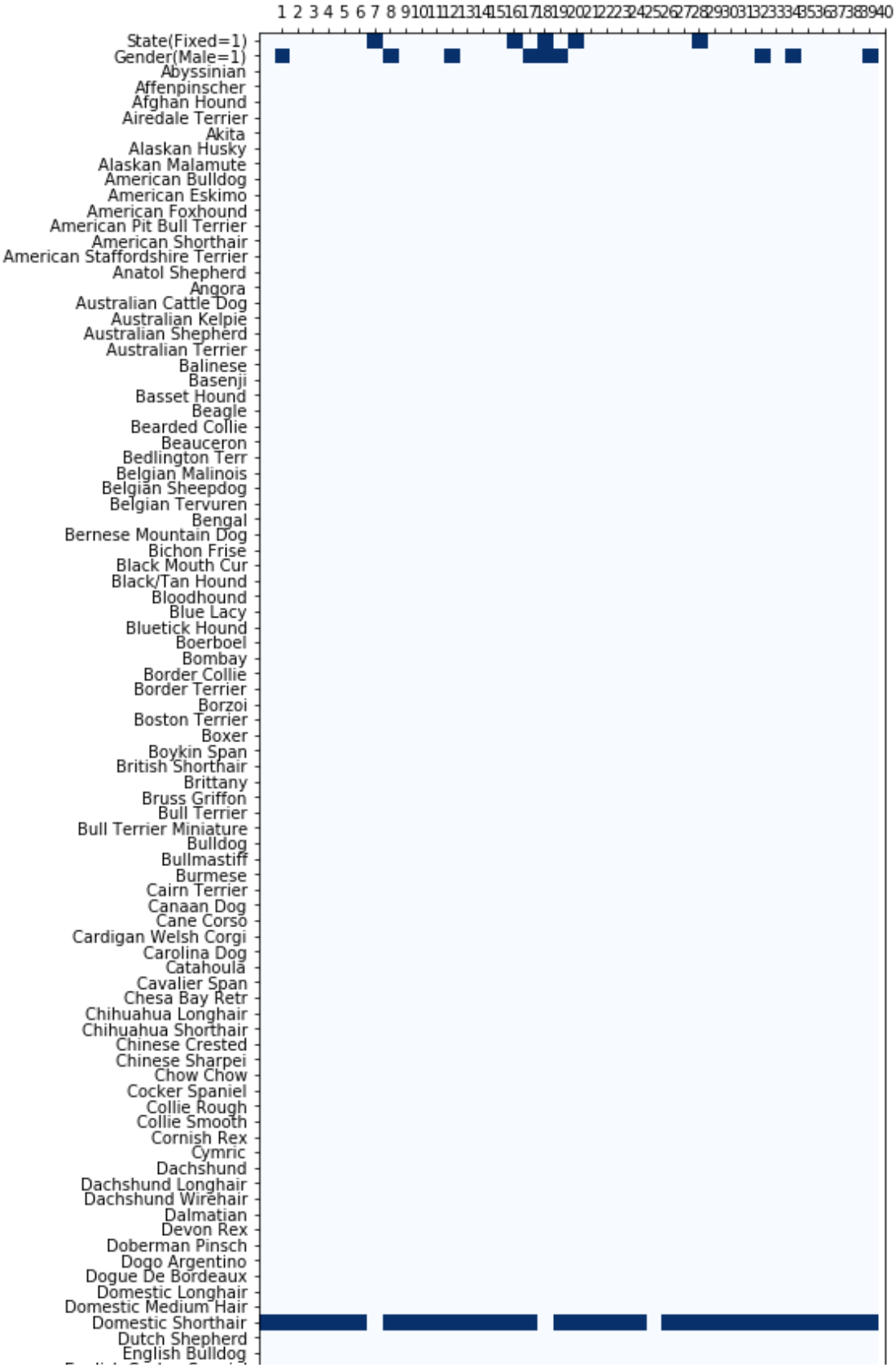

Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

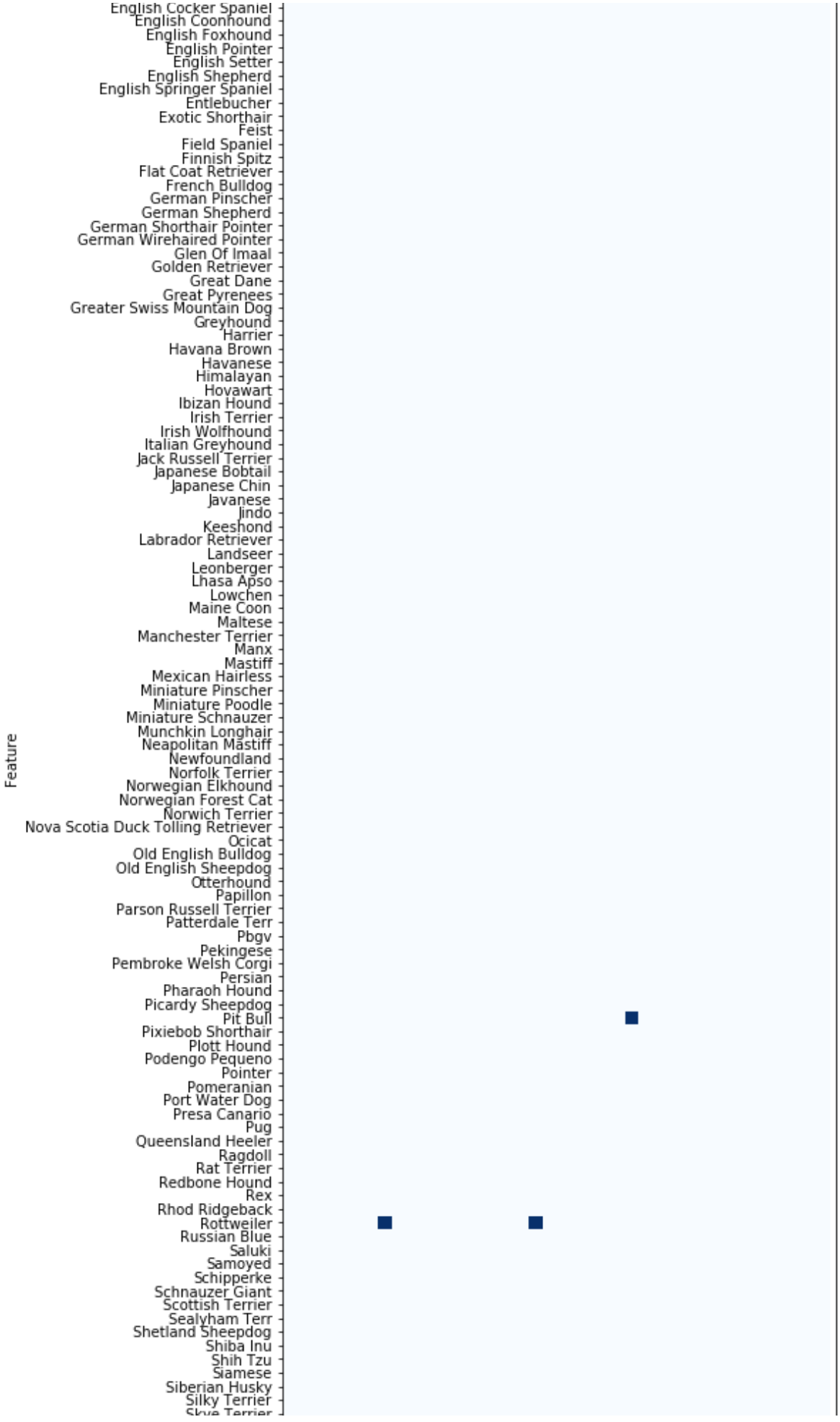
Transfer is predicted when the true value is Adoption: 265 times.

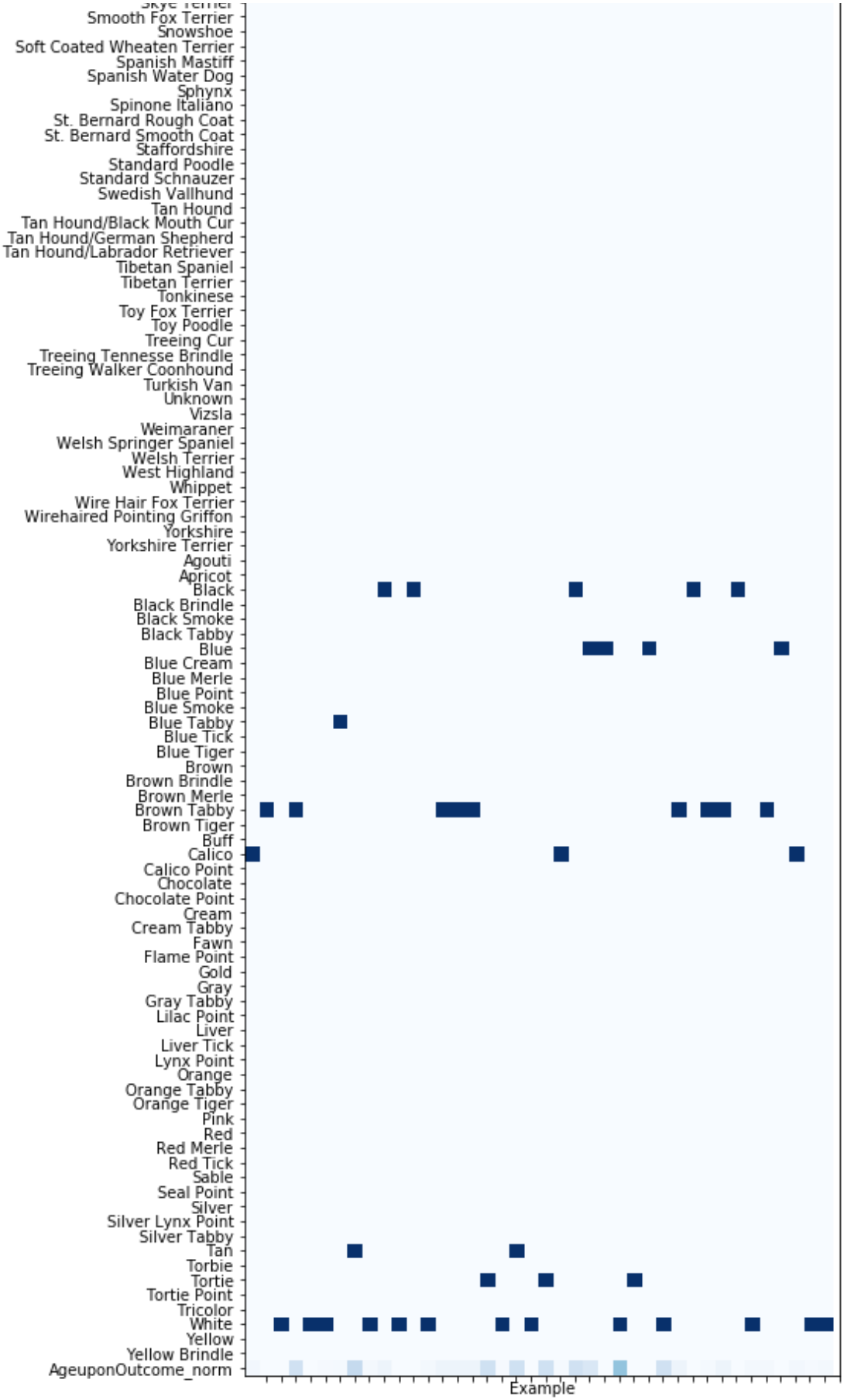

English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

Transfer is predicted when the true value is Euthanasia: 192 times.




Feature
English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Skye Terrier


Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

The plots of error examples are at the end of the above error analysis output.

We evaluated the most common types of errors. In fact, on our first pass, the error analysis helped us identify that our AgeuponOutcome variable was not split finely enough. We went back and simply normalized the age, in weeks, and this improved our performance by almost 6%. (The first pass is not shown in our notebook.)

The error analysis, however, is difficult to interpret, due to the sparsity of our dataset. The trends in the common mistakes are difficult to decipher. Very common breeds, such as domestic shorthairs, seem to be mispredicted as “transfer” or “died”, when the actual label was frequently euthanasia; this was especially common for cats. As the domestic shorthair breed appears for both cats and dogs, our model was getting confused on the prediction class. We briefly looked at the effect of adding interaction features in our models which can be seen in the code below (i.e. domestic shorthair * animal type) for further improvements, but we decided not to include this feature, as we saw no significant improvement.

```
train_data_pd = train_df_binarization.iloc[: , 13:] train_labels_pd =
train_df_binarization['OutcomeType'] train_data_pd['AnimalType_x_DomesticShorthair'] =
>train_data_pd['AnimalType_Binarize(Dog=1)'] train_data_pd['Domestic Shorthair']
train_data_pd['AnimalType_x_White'] = train_data_pd['AnimalType_Binarize(Dog=1)']
>train_data_pd['White'] train_data_pd.head()
```

Test Performance and Predict on the Test Data

Now that we have satsifactorily optimized our random forest classifier and conducted error analysis on the outcome classes where our model was not performing at its best, we will test our performance. We decided to test our performance by evaluating the log loss function of our classifiers. Log loss analyzes the accuracy of our random forest classifier, as it penalizes false predictions, and it is the measure used in the Kaggle competition. Our goal here will be to minimize the log loss function, or at least decrease it substantially with respect to our dummy classifier.

Here, we will test the winning algorithm and show the best performance we have been able to achieve, as well as evaluate our log loss function.

In [90]:
```
# Show the f1_dict, where we have been storing f1 values.
f1_dict
```

Out[90]:
```
{'Dumb': ['0.40', '0.23', ''],
 'Logistic Regression with Raw Data': ['0.63', '0.59', ''],
 'Decision Tree with Raw Data': ['0.59', '0.58', ''],
 'Random Forest with Raw Data': ['0.60', '0.58', ''],
 'XGBoost with Raw Data': ['0.63', '0.61', ''],
 'Logistic Regression with Oversampled Data': ['0.50', '0.47', ''],
 'Random Forest with Oversampled Data': ['0.69', '0.69', ''],
 'XGBoost with Oversampled Data': ['0.53', '0.52', ''],
 'Random Forest with Undersampled Data': ['0.39', '0.38', ''],
 'Multinomial Naive Bayes with Oversampled Data': ['0.40', '0.35', ''],
 'Multinomial Naive Bayes with Raw Data': ['0.52', '0.50', ''],
 'Random Forest with Oversampled, Projected Data': ['0.69', '0.68', '']}
```

In [91]:
```
# What is the log loss of the "dumb" classifier that predicts all adoption?

# Create numpy array, pred_prob, that predicts all adoption.
pred_prob = np.zeros((len(train_labels),5))
pred_prob[:, 0] = 1

# Calculate and print log_loss metric for the dumb classifier.
log_loss_metric = log_loss(train_labels, pred_prob
                           , labels=['Adoption'
                                     , 'Died'
                                     , 'Euthanasia'
                                     , 'Return_to_owner'
                                     , 'Transfer'
                                    ]
                          )
f1_dict['Dumb'][2] = '{:.2f}'.format(log_loss_metric)
print('Log loss for the dumb classifier is {:.2f}.'.format(log_loss_metric))
```

```
Log loss for the dumb classifier is 20.62.
```

In [92]:
```
# Pick our best classifier from above.
clf_optimal = RandomForestClassifier(max_depth=60, min_samples_split=4, criter
ion='gini')

# Fit it.
clf_optimal.fit(X_train, y_train)

# Predict probabilities.
pred_prob = clf_optimal.predict_proba(X_test)

# Calculate and print log_loss metric.
log_loss_metric = log_loss(y_test, pred_prob
                           , labels=['Adoption'
                                     , 'Died'
                                     , 'Euthanasia'
                                     , 'Return_to_owner'
                                     , 'Transfer'
                                    ]
                          )
print('Log loss for our best classifier is: {:.2f}'.format(log_loss_metric))
```

```
Log loss for our best classifier is: 1.69
```

In [93]:
```
clfs = [clf_LR, clf_DT, clf_RF, clf_XGB]
logs = []

def logloss(clf):

    # Select classifier.
    clf = clf

    # Fit it
    clf.fit(X_train, y_train)

    # Predict probabilities.
    pred_prob = clf.predict_proba(X_test)

    # Calculate and print log_loss metric.
    log_loss_metric = log_loss(y_test, pred_prob
                               , labels=['Adoption'
                                         , 'Died'
                                         , 'Euthanasia'
                                         , 'Return_to_owner'
                                         , 'Transfer'
                                        ]
                              )
    logs.append('{:.2f}'.format(log_loss_metric))
    print('Log loss: {:.2f}'.format(log_loss_metric))

for clf in clfs:
    logloss(clf)
```

```
Log loss: 1.24
Log loss: 3.45
Log loss: 1.42
Log loss: 1.10
```

The log loss of our "dumb" classifier that predicts all shelter animals to fall in the majority class of adoption is 20.62, showing a high degree of inaccuracy in its predictions. Our random forest classifier with the best parameters has a log loss of 1.62; a huge improvement compared to the dumb classifier! We also evaluated the log loss of each of the four classifiers that we have been working with: logistic regression, decision trees, random forests, and XGBoost. The logistic regression and XGBoost had a slightly lower log loss output than the random forest classifier. However, we have to weigh the combination of log loss and weighted f1-score when thinking about the predictive power of an algorithm.

In [94]:
```
# Append log loss values to scoring dictionary.
f1_dict['Logistic Regression with Raw Data'][2] = logs[0]
f1_dict['Decision Tree with Raw Data'][2] = logs[1]
f1_dict['Random Forest with Raw Data'][2] = logs[2]
f1_dict['XGBoost with Raw Data'][2] = logs[3]
```

In [95]:
```
# Make data frame from scoring dictionary.
df = pd.DataFrame(f1_dict)
df = df.transpose()
df.columns = ['Accuracy', 'F1 Score', 'Log Loss']
df
```

Out[95]:

	Accuracy	F1 Score	Log Loss
Decision Tree with Raw Data	0.59	0.58	3.45
Dumb	0.40	0.23	20.62
Logistic Regression with Oversampled Data	0.50	0.47	
Logistic Regression with Raw Data	0.63	0.59	1.24
Multinomial Naive Bayes with Oversampled Data	0.40	0.35	
Multinomial Naive Bayes with Raw Data	0.52	0.50	
Random Forest with Oversampled Data	0.69	0.69	
Random Forest with Oversampled, Projected Data	0.69	0.68	
Random Forest with Raw Data	0.60	0.58	1.42
Random Forest with Undersampled Data	0.39	0.38	
XGBoost with Oversampled Data	0.53	0.52	
XGBoost with Raw Data	0.63	0.61	1.10

Before submitting our test data class labels, we will perform the same PCA from our training data on our test data for feature congruence. As our test data does not have labeled outcome classes, our submission file will simply show the predicted classes. Our Kaggle competition is fairly old, so we will not be able to have our predicted outcome classes on the test data graded.

In [96]:
```
# Project test data to 30 principal components.

# Fit principal component analysis, using PCA from above.
print(pca)
projected_X_test = pca.transform(test_data)

# Predict test labels and probabilities.
pred_prob= clf_optimal.predict_proba(projected_X_test)
pred_labels = clf_optimal.predict(projected_X_test)
print(pred_prob)
print(pred_labels)

# Convert the dataset of predicted probabilities to a dataframe.
pred_prob = pd.DataFrame(data = pred_prob,
                         columns = ['Adoption', 'Died', 'Euthanasia', 'Return_
to_owner', 'Transfer'])

# Convert the dataset of predicted labels of the maximum probability to a data
frame.
pred_labels = pd.DataFrame(data = pred_labels,
                           columns = ['Max_Prob_Label'])

# Join the two dataframes, writing to a CSV file for submission.
prob_submission = pred_labels.join(pred_prob)
prob_submission.to_csv('data/prob_submission.csv')
```

```
PCA(copy=True, iterated_power='auto', n_components=30, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)
[[0.         0.         0.06       0.03333333 0.90666667]
 [0.625      0.         0.         0.3        0.075     ]
 [0.24160864 0.         0.14069433 0.19797973 0.41971729]
 ...
 [0.         0.         0.09908068 0.         0.90091932]
 [0.8        0.         0.         0.2        0.        ]
 [0.13333333 0.         0.2        0.17428571 0.49238095]]
['Transfer' 'Adoption' 'Transfer' ... 'Transfer' 'Adoption' 'Transfer']
```

Now we have a CSV file containing the probability of each outcome for each observation. We prefer the output showing probabilities over straight class predictions because it helps communicate the level of uncertainty in the predictions. In the left most column of the CSV file, we have shown the class with the highest predicted probability for reference. A sample of the CSV file is shown below.

In [97]:
```
prob_submission.head()
```

Out[97]:

	Max_Prob_Label	Adoption	Died	Euthanasia	Return_to_owner	Transfer
0	Transfer	0.000000	0.0	0.060000	0.033333	0.906667
1	Adoption	0.625000	0.0	0.000000	0.300000	0.075000
2	Transfer	0.241609	0.0	0.140694	0.197980	0.419717
3	Transfer	0.000000	0.0	0.000000	0.433333	0.566667
4	Adoption	0.528965	0.0	0.000000	0.392912	0.078123

Appendix:

What if we only predicted two class outcomes?

As we saw throughout our models, the died, euthanasia, and transfer outcome classes continually brought down our f1-scores. Interestingly, though, return to Owner was not one of the outcomes that reduced performance. This is counterintuitive, as, on the surface, return to owner seems as though it would be the most random outcome.

We adhered to the Kaggle competition guidance, which required predictions for all five outcomes, but as we navigated our models, we were left to wonder if a more suitable approach might exist. We noted some problems with the original five outcome classes. Logically, some of the outcomes appeared to be somewhat random, particularly return to owner and, to some degree, transfer.

To explore this, we decided to create a model that only had two classes: positive and negative. The positive classes are those in which the shelter animal leaves the system (i.e. adoption and return to owner), while the negative classes are those in which the animal leaves the system in a negative way or remains in the system (i.e. died, euthanasia, and transfer). We examine an updated dataset and models based on this approach below.

Creating the model

In [98]:
```
def binarize_outcome(x):
    if x['OutcomeType'] == 'Adoption': return 'Positive'
    elif x['OutcomeType'] == 'Died': return 'Negative'
    elif x['OutcomeType'] == 'Euthanasia': return 'Negative'
    elif x['OutcomeType'] == 'Return_to_owner': return 'Positive'
    elif x['OutcomeType'] == 'Transfer': return 'Negative'


# Grab all outcomes.
train_labels_pd_bin = train_df_binarize['OutcomeType']
train_labels_pd_bin = train_df_binarize.apply(binarize_outcome, axis=1)
train_labels_pd_bin.head()
```

Out[98]:
```
0    Positive
1    Negative
2    Positive
3    Negative
4    Negative
dtype: object
```

In [99]:
```
# We need a numpy array to run the models efficiently, so we convert all data
 here.
train_labels_bin = train_labels_pd_bin.values

# Check train_data and train_labels shape.
print('train_labels_bin shape:', train_labels_bin.shape)
```

```
train_labels_bin shape: (26729,)
```

Working and Running the Model

First, we look at the performance of the 'dumb' classifier, which assigns all outcomes to the majority class: positive.

In [100]:
```
# Set all predicted values to "Adoption".
pred_labels_bin = train_labels_bin.copy()
pred_labels_bin.fill('Positive')
print(classification_report(train_labels_bin, pred_labels_bin))
print('Accuracy = {:.2f}'.format(np.mean(train_labels_bin == pred_labels_bin
)))
```

```
              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00     11174
    Positive       0.58      1.00      0.74     15555

   micro avg       0.58      0.58      0.58     26729
   macro avg       0.29      0.50      0.37     26729
weighted avg       0.34      0.58      0.43     26729

Accuracy = 0.58
```

The weighted f1-score for the dumb classifier is 0.43, which is not great.

For this model, we will first examine the plain vanilla Logistic Regression classifier to baseline our results.

In [101]:
```
# Check fit of logistic regression.
clf_LR = LogisticRegression()
stratfit(clf_LR, train_data, train_labels_bin, n_splits=5)
```

TRAIN: 21383 and TEST: 5346

For train data:

	precision	recall	f1-score	support
Negative	0.86	0.62	0.72	8939
Positive	0.77	0.93	0.84	12444
micro avg	0.80	0.80	0.80	21383
macro avg	0.82	0.77	0.78	21383
weighted avg	0.81	0.80	0.79	21383

For test data:

	precision	recall	f1-score	support
Negative	0.86	0.64	0.73	2235
Positive	0.78	0.92	0.85	3111
micro avg	0.80	0.80	0.80	5346
macro avg	0.82	0.78	0.79	5346
weighted avg	0.81	0.80	0.80	5346

TRAIN: 21383 and TEST: 5346

For train data:

	precision	recall	f1-score	support
Negative	0.86	0.63	0.72	8939
Positive	0.77	0.93	0.84	12444
micro avg	0.80	0.80	0.80	21383
macro avg	0.82	0.78	0.78	21383
weighted avg	0.81	0.80	0.79	21383

For test data:

	precision	recall	f1-score	support
Negative	0.87	0.63	0.73	2235
Positive	0.78	0.93	0.85	3111
micro avg	0.80	0.80	0.80	5346
macro avg	0.82	0.78	0.79	5346
weighted avg	0.81	0.80	0.80	5346

TRAIN: 21383 and TEST: 5346

For train data:

	precision	recall	f1-score	support
Negative	0.86	0.63	0.73	8939
Positive	0.78	0.93	0.84	12444
micro avg	0.80	0.80	0.80	21383
macro avg	0.82	0.78	0.78	21383
weighted avg	0.81	0.80	0.79	21383

For test data:

	precision	recall	f1-score	support
Negative	0.86	0.62	0.72	2235
Positive	0.77	0.93	0.84	3111
micro avg	0.80	0.80	0.80	5346
macro avg	0.81	0.77	0.78	5346
weighted avg	0.81	0.80	0.79	5346

TRAIN: 21383 and TEST: 5346

For train data:

	precision	recall	f1-score	support
Negative	0.86	0.62	0.72	8939
Positive	0.77	0.93	0.84	12444
micro avg	0.80	0.80	0.80	21383
macro avg	0.82	0.78	0.78	21383
weighted avg	0.81	0.80	0.79	21383

For test data:

	precision	recall	f1-score	support
Negative	0.86	0.63	0.72	2235
Positive	0.78	0.92	0.84	3111
micro avg	0.80	0.80	0.80	5346
macro avg	0.82	0.78	0.78	5346
weighted avg	0.81	0.80	0.79	5346

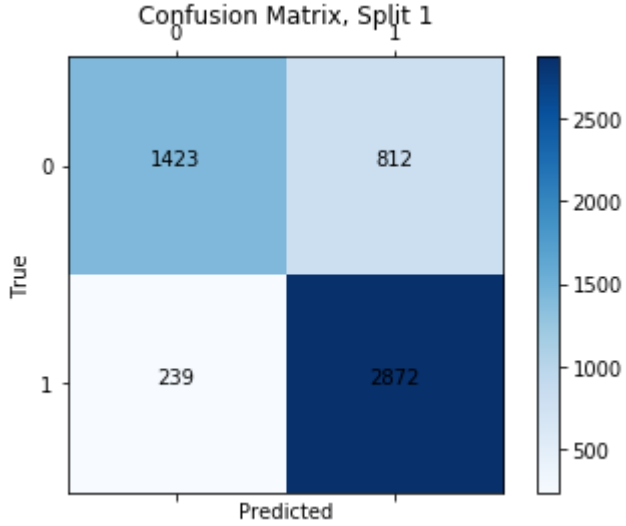
TRAIN: 21384 and TEST: 5345

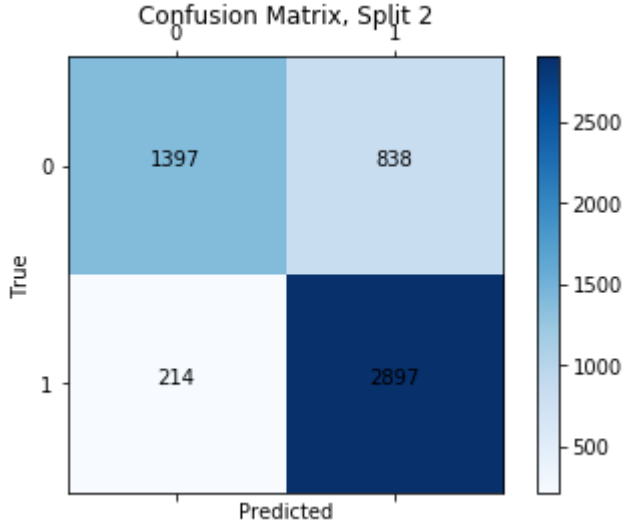
For train data:

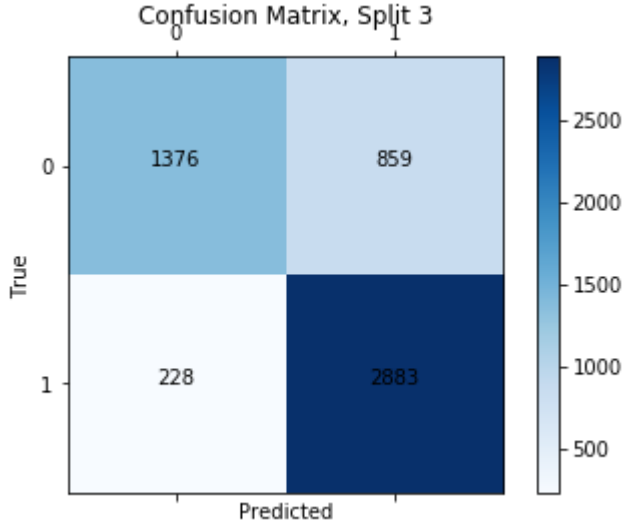
	precision	recall	f1-score	support
Negative	0.86	0.63	0.73	8940
Positive	0.78	0.93	0.84	12444
micro avg	0.80	0.80	0.80	21384
macro avg	0.82	0.78	0.78	21384
weighted avg	0.81	0.80	0.79	21384

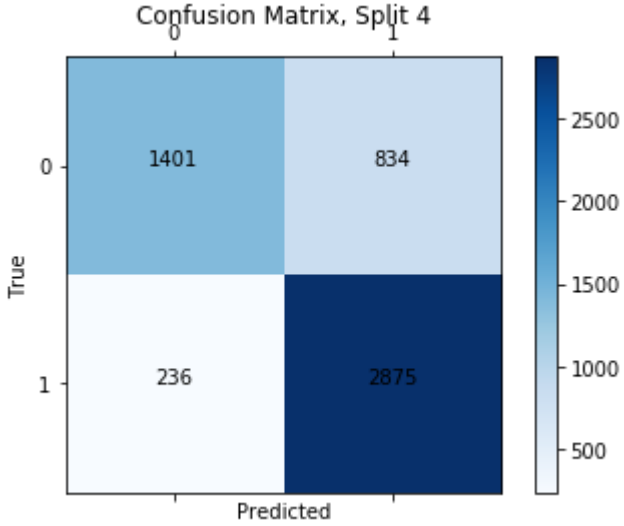
For test data:

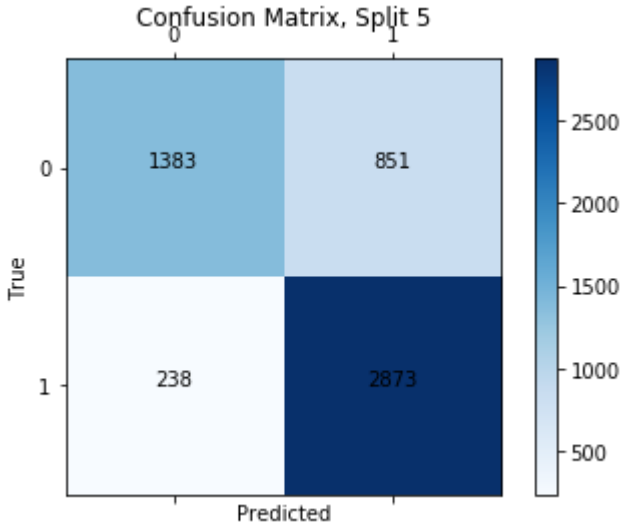
	precision	recall	f1-score	support
Negative	0.85	0.62	0.72	2234
Positive	0.77	0.92	0.84	3111
micro avg	0.80	0.80	0.80	5345
macro avg	0.81	0.77	0.78	5345
weighted avg	0.81	0.80	0.79	5345


Confusion Matrix, Split 1
0
1
0
1
True
1423
812
239
2872
Predicted
2500
2000
1500
1000
500


Confusion Matrix, Split 2
0
1
0
1
True
1397
838
214
2897
Predicted
2500
2000
1500
1000
500


Confusion Matrix, Split 3
0
1
0
1
True
1376
859
228
2883
Predicted
2500
2000
1500
1000
500





Our initial pass with out-of-the-box logistic regression performed well. The weighted f1-scores for both training and test folds were high, at 0.79.

Out of curiosity, we will do a short optimization exercise, but will only pull in the best cases of the model above for other items to evaluate a consistent comparison. Therefore, we will oversample our data and use a random forest classifier throughout.

As we are creating a new oversampled dataset (to properly weigh our two outcome classes), we have to repeat our PCA analysis as well. However, we want to keep this model similar to our prior model. Therefore, we will be keeping the same number of PCA components as we determined to be optimal above.

In [102]:
```
# Copying the training data to oversample with only two labels.
train_data_bin = np.copy(train_data)

# Oversample to balance outcome classes.
ros = RandomOverSampler(random_state=0)
train_data_bin_os, train_labels_bin_os = ros.fit_resample(train_data_bin, trai
n_labels_bin)

# Count to double check how many animals are in each class after we employ the
RandomOverSampler.
print(sorted(Counter(train_labels_bin_os).items()))
```

```
[('Negative', 15555), ('Positive', 15555)]
```

In [103]:
```
# Set the number of principal components.
pca = PCA(n_components=30)

# Fit and transform based on train data, but oversampled.
projected_X_train_bin_os = pca.fit_transform(train_data_bin_os)
```

In [104]:
```
# This code optimizes a random forest classifier.
clf = RandomForestClassifier()

param = {'max_depth': np.arange(20,120,20)
         , 'min_samples_split': np.arange(4,10,2)
         # , 'n_estimators': np.arange(10, 120, 30)
         , 'criterion': ['gini', 'entropy']}

# Optimize classifier, use f1_weighting, use 5 StratifiedKFolds.
clf_best_param = GridSearchCV(clf, param_grid = param, cv=5, scoring = 'f1_wei
ghted')

# Fit optimized classifier.
clf_best = clf_best_param.fit(projected_X_train_bin_os, train_labels_bin_os)

# Predict on regular data.
# Predict values.
pred_labels = clf_best_param.predict(projected_X_train_bin_os)

print(clf_best.best_params_)
```

```
{'criterion': 'entropy', 'max_depth': 20, 'min_samples_split': 6}
```

In [105]:
```
# Run random forest with the best parameters.
clf_RF = RandomForestClassifier(max_depth=40, min_samples_split=6, criterion=
'gini')
stratfit(clf_RF, projected_X_train_bin_os, train_labels_bin_os, n_splits=5)
```

TRAIN: 24888 and TEST: 6222

For train data:

	precision	recall	f1-score	support
Negative	0.89	0.83	0.86	12444
Positive	0.84	0.90	0.87	12444
micro avg	0.86	0.86	0.86	24888
macro avg	0.87	0.86	0.86	24888
weighted avg	0.87	0.86	0.86	24888

For test data:

	precision	recall	f1-score	support
Negative	0.80	0.74	0.77	3111
Positive	0.76	0.81	0.78	3111
micro avg	0.78	0.78	0.78	6222
macro avg	0.78	0.78	0.78	6222
weighted avg	0.78	0.78	0.78	6222

TRAIN: 24888 and TEST: 6222

For train data:

	precision	recall	f1-score	support
Negative	0.88	0.84	0.86	12444
Positive	0.85	0.89	0.87	12444
micro avg	0.86	0.86	0.86	24888
macro avg	0.86	0.86	0.86	24888
weighted avg	0.86	0.86	0.86	24888

For test data:

	precision	recall	f1-score	support
Negative	0.79	0.76	0.78	3111
Positive	0.77	0.80	0.79	3111
micro avg	0.78	0.78	0.78	6222
macro avg	0.78	0.78	0.78	6222
weighted avg	0.78	0.78	0.78	6222

TRAIN: 24888 and TEST: 6222

For train data:

	precision	recall	f1-score	support
Negative	0.88	0.83	0.86	12444
Positive	0.84	0.89	0.87	12444
micro avg	0.86	0.86	0.86	24888
macro avg	0.86	0.86	0.86	24888
weighted avg	0.86	0.86	0.86	24888

For test data:

	precision	recall	f1-score	support
Negative	0.80	0.76	0.78	3111
Positive	0.77	0.81	0.79	3111
micro avg	0.79	0.79	0.79	6222
macro avg	0.79	0.79	0.79	6222
weighted avg	0.79	0.79	0.79	6222

TRAIN: 24888 and TEST: 6222

For train data:

	precision	recall	f1-score	support
Negative	0.89	0.83	0.86	12444
Positive	0.84	0.90	0.87	12444
micro avg	0.86	0.86	0.86	24888
macro avg	0.87	0.86	0.86	24888
weighted avg	0.87	0.86	0.86	24888

For test data:

	precision	recall	f1-score	support
Negative	0.80	0.73	0.77	3111
Positive	0.75	0.82	0.78	3111
micro avg	0.78	0.78	0.78	6222
macro avg	0.78	0.78	0.78	6222
weighted avg	0.78	0.78	0.78	6222

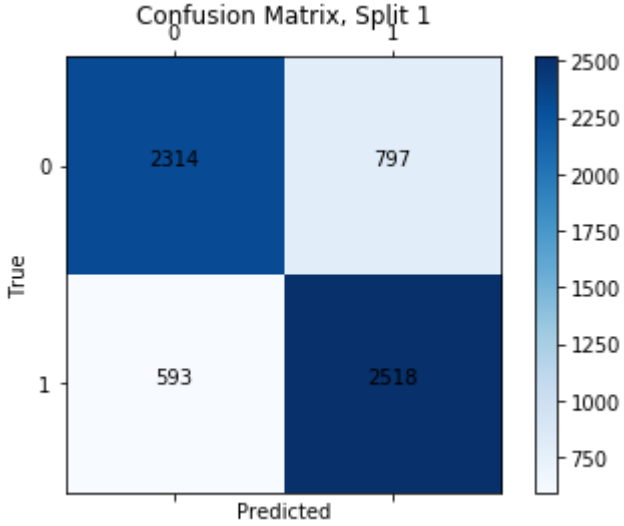
TRAIN: 24888 and TEST: 6222

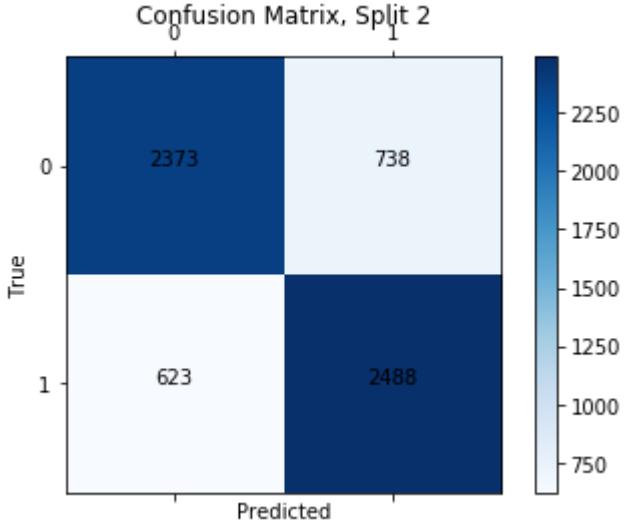
For train data:

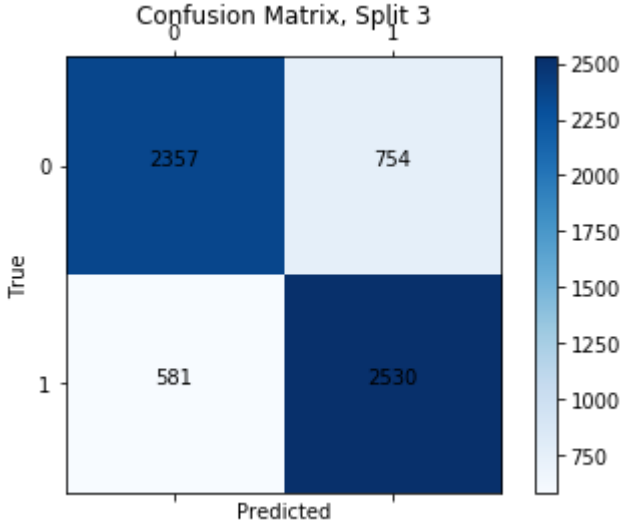
	precision	recall	f1-score	support
Negative	0.89	0.83	0.86	12444
Positive	0.84	0.90	0.87	12444
micro avg	0.86	0.86	0.86	24888
macro avg	0.87	0.86	0.86	24888
weighted avg	0.87	0.86	0.86	24888

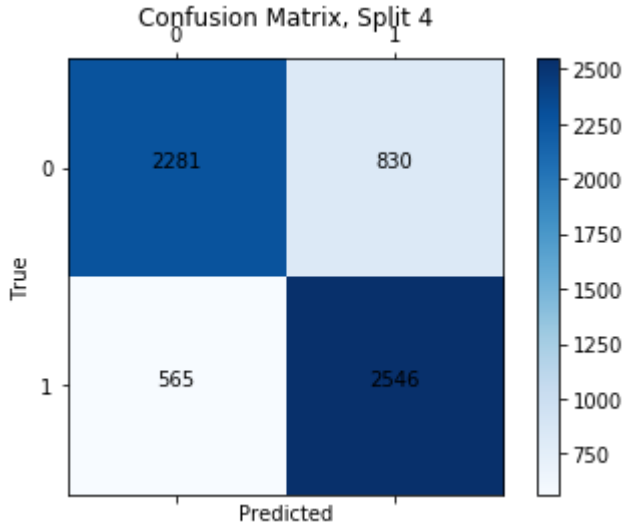
For test data:

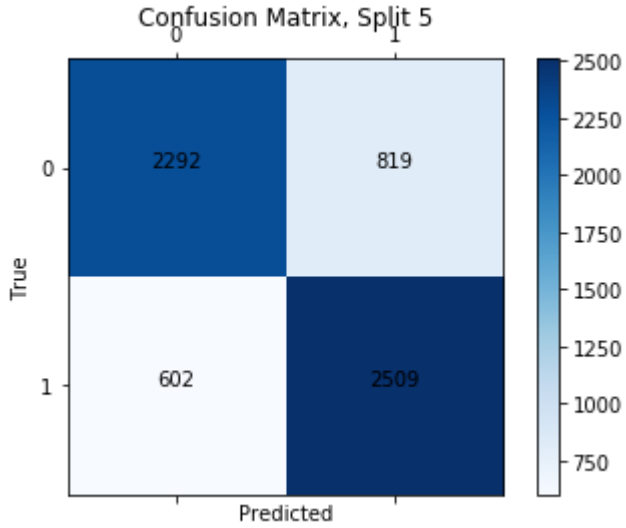
	precision	recall	f1-score	support
Negative	0.79	0.74	0.76	3111
Positive	0.75	0.81	0.78	3111
micro avg	0.77	0.77	0.77	6222
macro avg	0.77	0.77	0.77	6222
weighted avg	0.77	0.77	0.77	6222


Confusion Matrix, Split 1
0
1
2314
797
593
2518
True
Predicted
2500
2250
2000
1750
1500
1250
1000
750


Confusion Matrix, Split 2
0
1
2373
738
623
2488
True
Predicted
2250
2000
1750
1500
1250
1000
750


Confusion Matrix, Split 3
0
1
2357
754
581
2530
True
Predicted
2500
2250
2000
1750
1500
1250
1000
750





We can see that the out-of-the-box logistic regression classifier marginally outperforms the class-balanced and PCA-projected random forest classifier. With binary outcomes, our classes are almost balanced, so oversampling is not strictly necessary. However, we will continue using oversampled data for consistency.

Error Analysis for Binary Outcomes

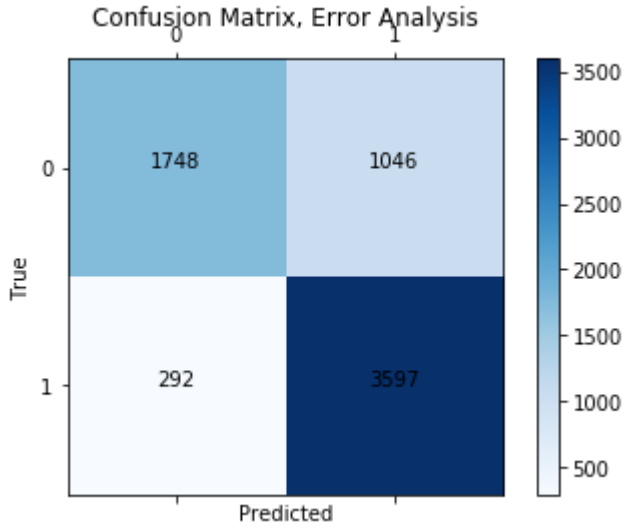
In [106]:
```
# Split training, projected data into training and test.
indices = np.asarray(range(len(train_data)))
X_train_bin, X_test_bin, y_train_bin, y_test_bin,indices_train_bin,indices_tes
t_bin= train_test_split(train_data

, train_labels_bin

, indices

, test_size = .25

, stratify = train_labels_bin

, random_state = 99)
print(X_train_bin.shape)
print(X_test_bin.shape)
print(y_train_bin.shape)
print(y_test_bin.shape)
print(indices_train_bin.shape)
print(indices_test_bin.shape)
```

```
(20046, 283)
(6683, 283)
(20046,)
(6683,)
(20046,)
(6683,)
```

In [107]:
```
# Run error analysis for LogisticRegression.
clf_LR = LogisticRegression()
clf_LR.fit(X_train_bin, y_train_bin)
pred_labels = clf_LR.predict(X_test_bin)

labels = ['Negative', 'Positive']
find_and_show_errors(fit_labels_true=y_test_bin, train_data=train_data_bin_os
                     , pred_labels=pred_labels,  indices_test=indices_test_bin
                     , labels=labels, N=2, J=40
                    )
```

Label Legend
0: Negative
1: Positive

Confusion Matrix



Top Errors:

Positive is predicted when the true value is Negative: 1046 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Blue Lacy
White
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Medium Hair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Labrador Retriever
White
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Miniature Pinscher
Red
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Chihuahua Shorthair
Fawn
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Shih Tzu

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Rat Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Great Pyrenees
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         American Pit Bull Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         Golden Retriever
         Gold
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         Black
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 15: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Siamese
         Blue Point
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Field Spaniel
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Calico
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         State(Fixed=1)
         Rat Terrier
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         White
```

```
         AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
         State(Fixed=1)
         Harrier
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Boxer
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Dachshund
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chesa Bay Retr
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
```

```
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Jack Russell Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pointer
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Domestic Longhair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chinese Sharpei
         Fawn
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Snowshoe
         White
```

        AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
        State(Fixed=1)
        Doberman Pinsch
        Tricolor
        AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
        State(Fixed=1)
        Gender(Male=1)
        Chihuahua Longhair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
        State(Fixed=1)
        Domestic Shorthair
        Tortie
        AgeuponOutcome_norm
All other features equal to 0.

Negative is predicted when the true value is Positive: 292 times.

Examples of this error:

Example 1: Non-zero features:
        Boxer
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
        Doberman Pinsch
        Tan
        AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
        German Shepherd
        Tan
        AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
        Yorkshire Terrier
        Brown
        AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
        Gender(Male=1)
        Chihuahua Shorthair

```
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
         Gender(Male=1)
         Miniature Poodle
         Gray
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         Rottweiler
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         Gender(Male=1)
         Rottweiler
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         Pit Bull
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         Gender(Male=1)
         Anatol Shepherd
         Cream
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         Gender(Male=1)
         German Shepherd
         Brown
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 14: Non-zero features:
         Gender(Male=1)
         German Shepherd
         Sable
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         German Shepherd
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         Pit Bull
         White
All other features equal to 0.

Example 17: Non-zero features:
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         Lhasa Apso
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         Gender(Male=1)
         Domestic Shorthair
         Cream Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         Miniature Poodle
         Buff
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
         Gender(Male=1)
         Miniature Pinscher
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 23: Non-zero features:
         Domestic Shorthair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         Gender(Male=1)
         Beagle
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         Gender(Male=1)
         Chow Chow
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         Pit Bull
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         Domestic Medium Hair
         Blue Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         Gender(Male=1)
         Chihuahua Longhair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         Gender(Male=1)
         Queensland Heeler
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         Domestic Shorthair
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         Rottweiler
         Brown
         AgeuponOutcome_norm
All other features equal to 0.
```

Example 32: Non-zero features:
Yorkshire Terrier
Red
AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
Dachshund
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
Gender(Male=1)
Golden Retriever
Gold
AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
Domestic Shorthair
Tortie
AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
Gender(Male=1)
Chow Chow
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
Domestic Medium Hair
White
AgeuponOutcome_norm
All other features equal to 0.

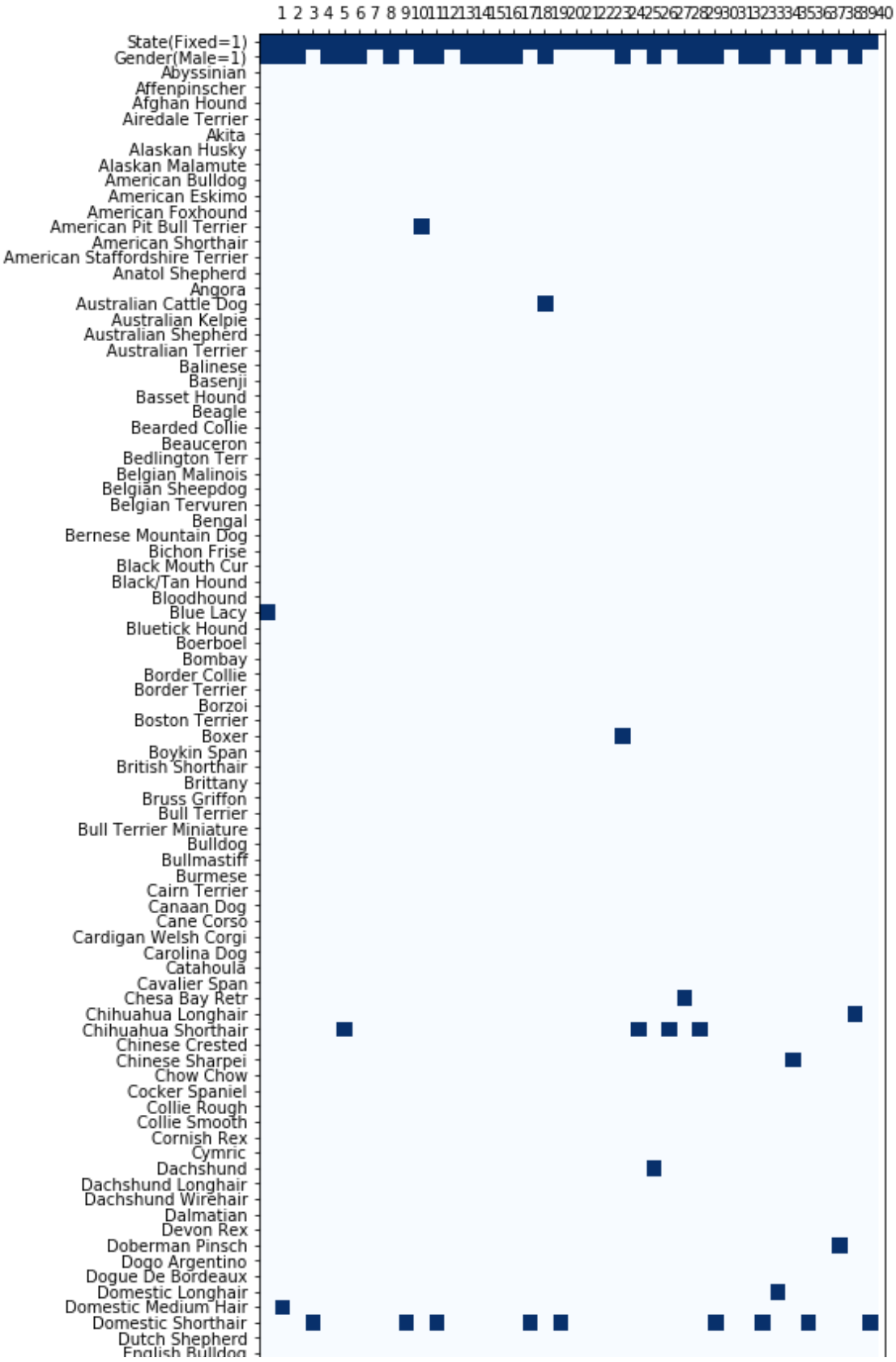
Example 38: Non-zero features:
Gender(Male=1)
Yorkshire Terrier
Tricolor
AgeuponOutcome_norm
All other features equal to 0.

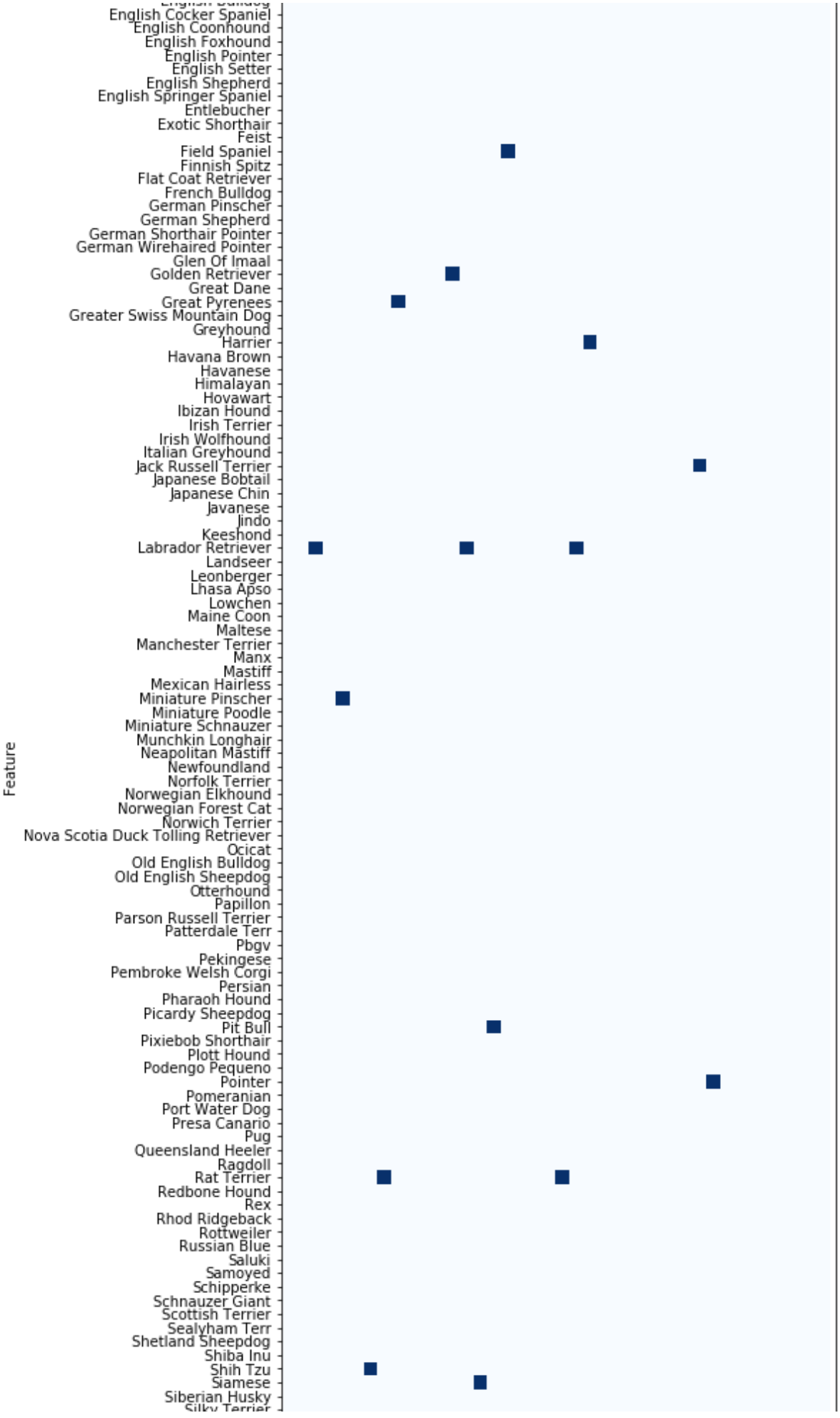
Example 39: Non-zero features:
Gender(Male=1)
Miniature Pinscher
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
Chihuahua Shorthair
Brown
AgeuponOutcome_norm

All other features equal to 0.

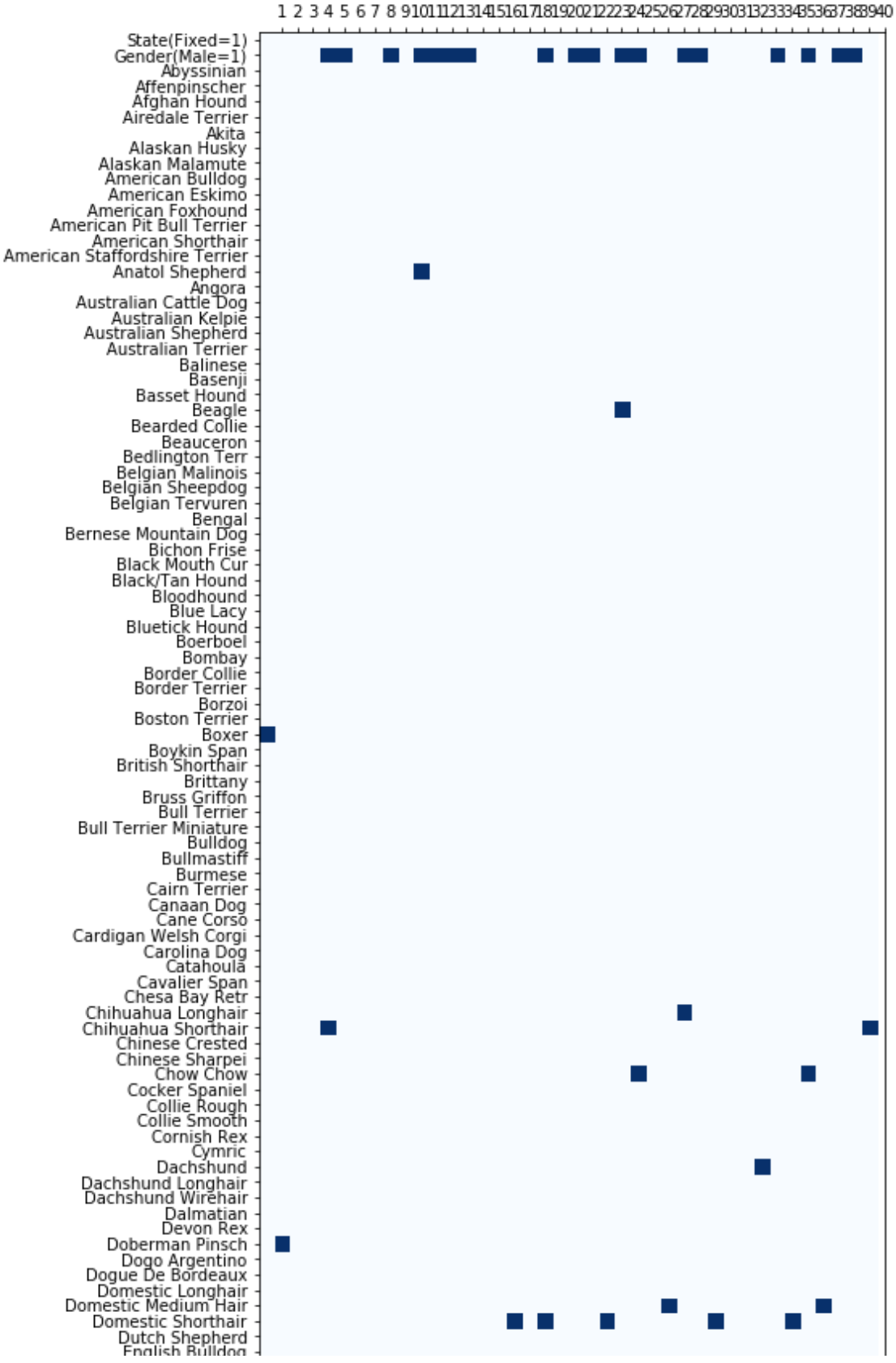
Positive is predicted when the true value is Negative: 1046 times.

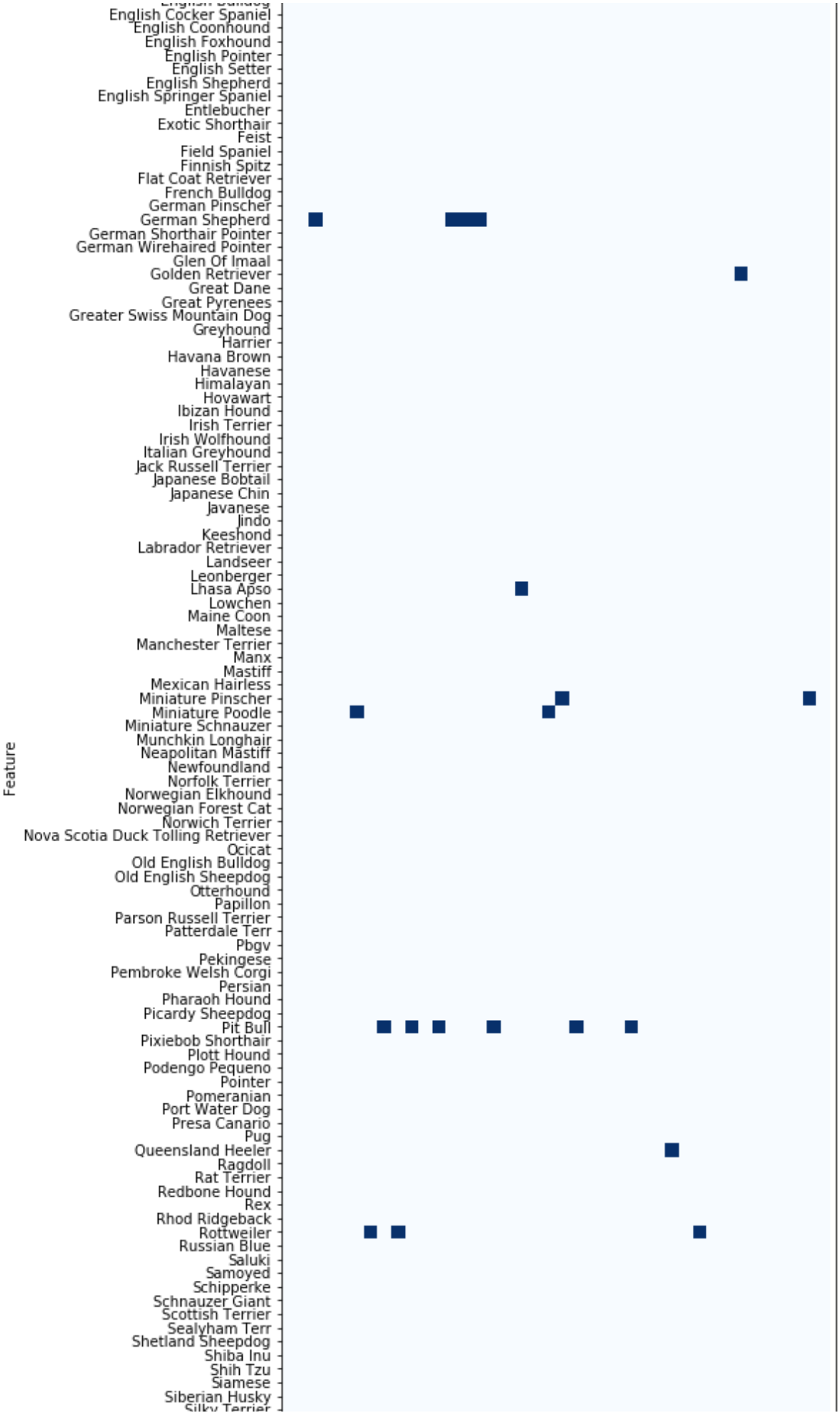


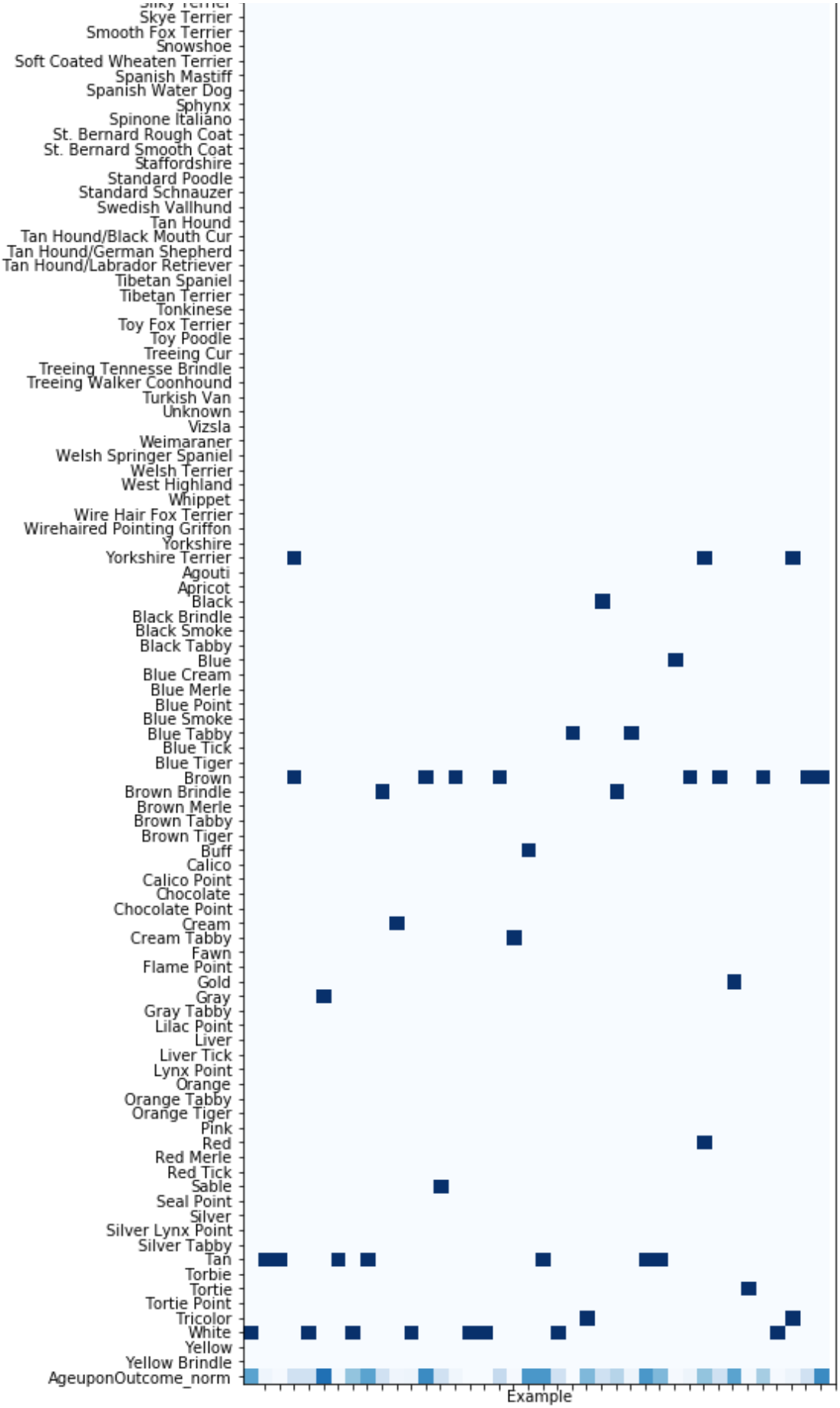

English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

Negative is predicted when the true value is Positive: 292 times.



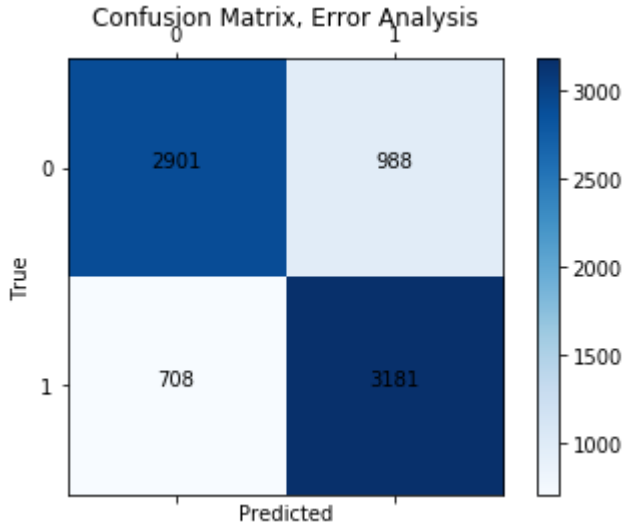

English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

In [108]:
```
# Run error analysis for random forest.
indices = np.asarray(range(len(projected_X_train_bin_os)))
X_train_bin, X_test_bin, y_train_bin, y_test_bin,indices_train_bin,indices_tes
t_bin= train_test_split(projected_X_train_bin_os

, train_labels_bin_os

, indices

, test_size = .25

, stratify = train_labels_bin_os

, random_state = 99)

clf_RF = RandomForestClassifier(max_depth=80, min_samples_split=6, criterion=
'entropy')
clf_RF.fit(X_train_bin, y_train_bin)
pred_labels = clf_RF.predict(X_test_bin)

labels = ['Negative', 'Positive']
find_and_show_errors(fit_labels_true=y_test_bin, train_data=train_data_bin_os
                     , pred_labels=pred_labels,  indices_test=indices_test_bin
                     , labels=labels, N=2, J=40
                    )
```

Label Legend
0: Negative
1: Positive

Confusion Matrix



Top Errors:

Positive is predicted when the true value is Negative: 988 times.

Examples of this error:

Example 1: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Pit Bull
White
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Pit Bull
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Chihuahua Shorthair
Black
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 6: Non-zero features:
State(Fixed=1)
Chihuahua Shorthair
Brown
AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
State(Fixed=1)
Chihuahua Shorthair
Brown
AgeuponOutcome_norm

```
All other features equal to 0.

Example 8: Non-zero features:
         Gender(Male=1)
         Miniature Poodle
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Rat Terrier
         Chocolate
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 13: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         German Shepherd
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
```

```
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
        State(Fixed=1)
        Gender(Male=1)
        Catahoula
        Brown
        AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
        State(Fixed=1)
        Gender(Male=1)
        Labrador Retriever
        Yellow
        AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
        Domestic Shorthair
        Silver Lynx Point
        AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
        Pit Bull
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
        State(Fixed=1)
        Gender(Male=1)
        Miniature Pinscher
        Brown
        AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
        State(Fixed=1)
        Domestic Shorthair
        Black
        AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
        State(Fixed=1)
        Beagle
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
        State(Fixed=1)
        Labrador Retriever
```

```
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
         Rottweiler
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Boxer
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
```

```
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Dachshund
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         State(Fixed=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Beagle
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 38: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Beagle
         Black
```

AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
State(Fixed=1)
Domestic Shorthair
Calico
AgeuponOutcome_norm
All other features equal to 0.

Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Siberian Husky
White
AgeuponOutcome_norm
All other features equal to 0.

Negative is predicted when the true value is Positive: 708 times.

Examples of this error:

Example 1: Non-zero features:
Domestic Medium Hair
Blue Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 2: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

Example 3: Non-zero features:
State(Fixed=1)
Chihuahua Shorthair
Red
AgeuponOutcome_norm
All other features equal to 0.

Example 4: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Medium Hair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 5: Non-zero features:
State(Fixed=1)
Domestic Shorthair
Orange Tabby
AgeuponOutcome_norm

```
All other features equal to 0.

Example 6: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         American Bulldog
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 7: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Chihuahua Shorthair
         Brown Brindle
         AgeuponOutcome_norm
All other features equal to 0.

Example 8: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Boston Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 9: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Boston Terrier
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 10: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Pit Bull
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 11: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Yorkshire Terrier
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 12: Non-zero features:
         Gender(Male=1)
         Bull Terrier
         White
         AgeuponOutcome_norm
All other features equal to 0.
```

```
Example 13: Non-zero features:
         Gender(Male=1)
         Pit Bull
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 14: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 15: Non-zero features:
         State(Fixed=1)
         Labrador Retriever
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 16: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Domestic Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 17: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Cream
         AgeuponOutcome_norm
All other features equal to 0.

Example 18: Non-zero features:
         German Shepherd
         Sable
         AgeuponOutcome_norm
All other features equal to 0.

Example 19: Non-zero features:
         State(Fixed=1)
         Domestic Medium Hair
         Black
         AgeuponOutcome_norm
All other features equal to 0.

Example 20: Non-zero features:
         State(Fixed=1)
         Domestic Shorthair
         Brown Tabby
         AgeuponOutcome_norm
All other features equal to 0.

Example 21: Non-zero features:
```

```
        State(Fixed=1)
        Domestic Shorthair
        Tortie
        AgeuponOutcome_norm
All other features equal to 0.

Example 22: Non-zero features:
        Gender(Male=1)
        Cocker Spaniel
        Black
        AgeuponOutcome_norm
All other features equal to 0.

Example 23: Non-zero features:
        State(Fixed=1)
        Catahoula
        Brown Merle
        AgeuponOutcome_norm
All other features equal to 0.

Example 24: Non-zero features:
        State(Fixed=1)
        Domestic Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 25: Non-zero features:
        State(Fixed=1)
        Gender(Male=1)
        Domestic Longhair
        Black
        AgeuponOutcome_norm
All other features equal to 0.

Example 26: Non-zero features:
        State(Fixed=1)
        Domestic Shorthair
        White
        AgeuponOutcome_norm
All other features equal to 0.

Example 27: Non-zero features:
        Chinese Sharpei
        Brown
        AgeuponOutcome_norm
All other features equal to 0.

Example 28: Non-zero features:
        State(Fixed=1)
        Domestic Medium Hair
        Tortie
        AgeuponOutcome_norm
All other features equal to 0.

Example 29: Non-zero features:
        Dachshund Wirehair
```

```
         Brown
         AgeuponOutcome_norm
All other features equal to 0.

Example 30: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         German Shepherd
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 31: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Australian Cattle Dog
         Tricolor
         AgeuponOutcome_norm
All other features equal to 0.

Example 32: Non-zero features:
         State(Fixed=1)
         Gender(Male=1)
         Great Dane
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 33: Non-zero features:
         Gender(Male=1)
         American Pit Bull Terrier
         Blue
         AgeuponOutcome_norm
All other features equal to 0.

Example 34: Non-zero features:
         Chihuahua Shorthair
         White
         AgeuponOutcome_norm
All other features equal to 0.

Example 35: Non-zero features:
         State(Fixed=1)
         Domestic Medium Hair
         Tortie
         AgeuponOutcome_norm
All other features equal to 0.

Example 36: Non-zero features:
         State(Fixed=1)
         Chihuahua Shorthair
         Tan
         AgeuponOutcome_norm
All other features equal to 0.

Example 37: Non-zero features:
         State(Fixed=1)
```

Gender(Male=1)
Beagle
Tricolor
AgeuponOutcome_norm
All other features equal to 0.

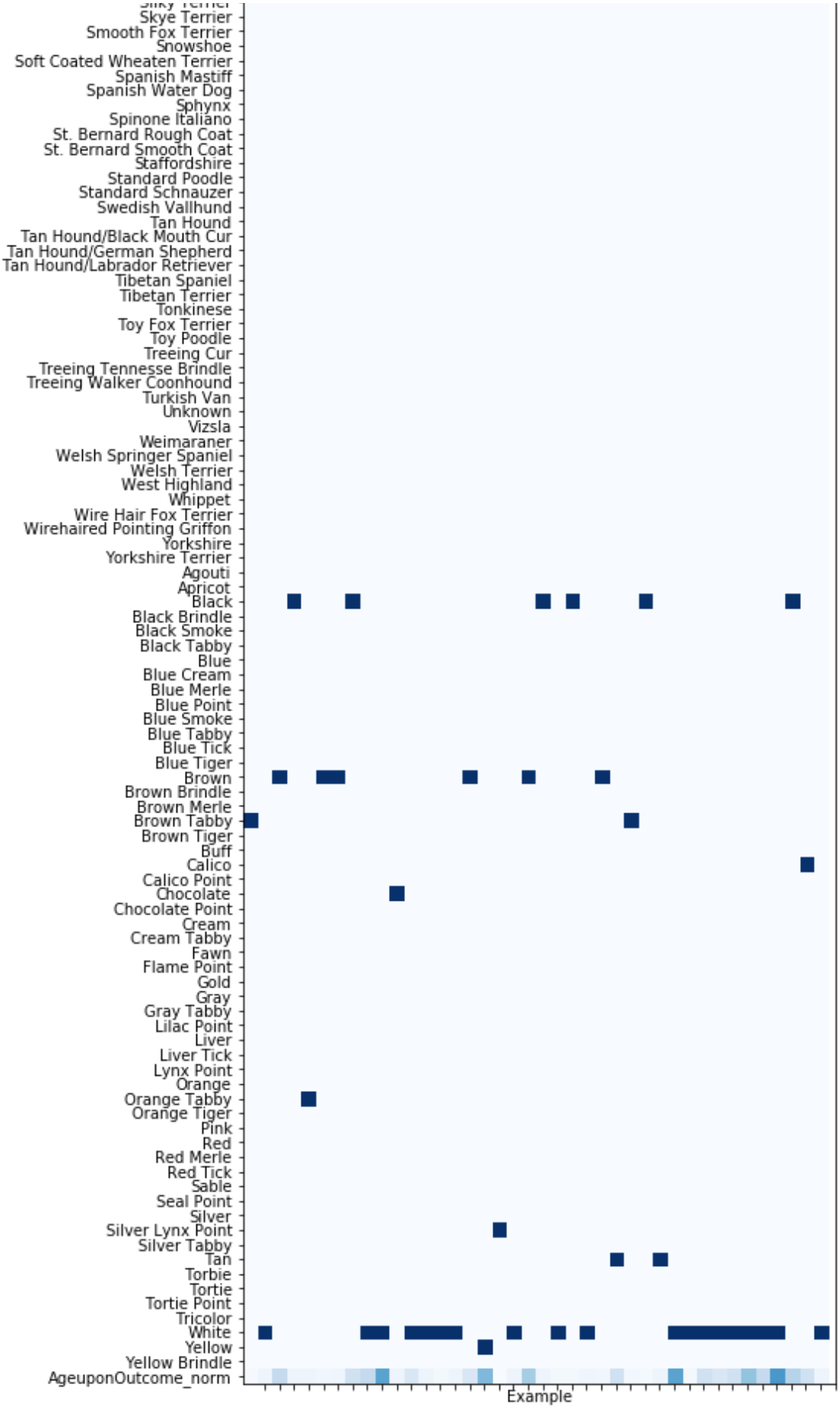
Example 38: Non-zero features:
Gender(Male=1)
Domestic Shorthair
White
AgeuponOutcome_norm
All other features equal to 0.

Example 39: Non-zero features:
Gender(Male=1)
Patterdale Terr
White
AgeuponOutcome_norm
All other features equal to 0.

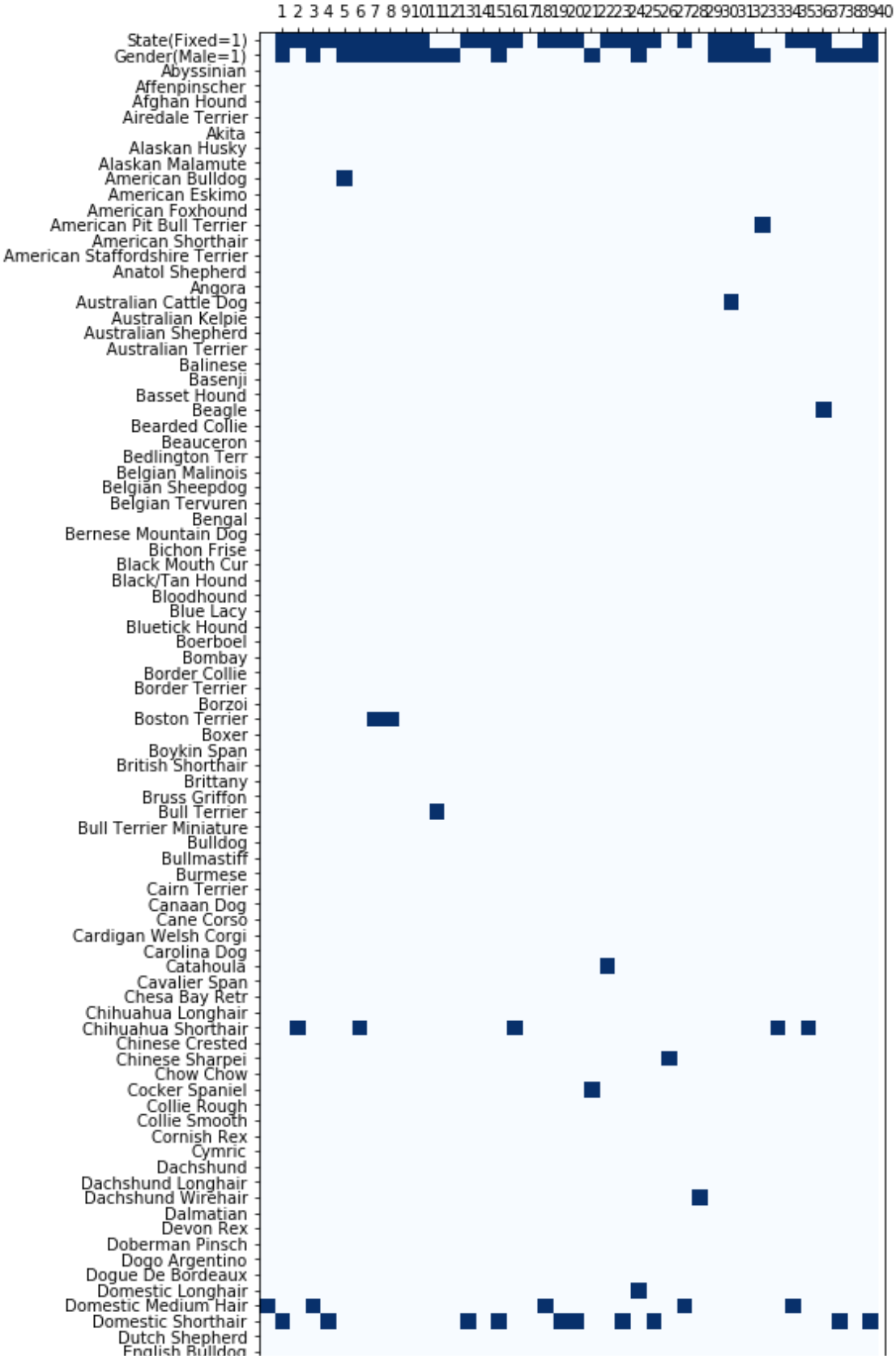
Example 40: Non-zero features:
State(Fixed=1)
Gender(Male=1)
Domestic Shorthair
Brown Tabby
AgeuponOutcome_norm
All other features equal to 0.

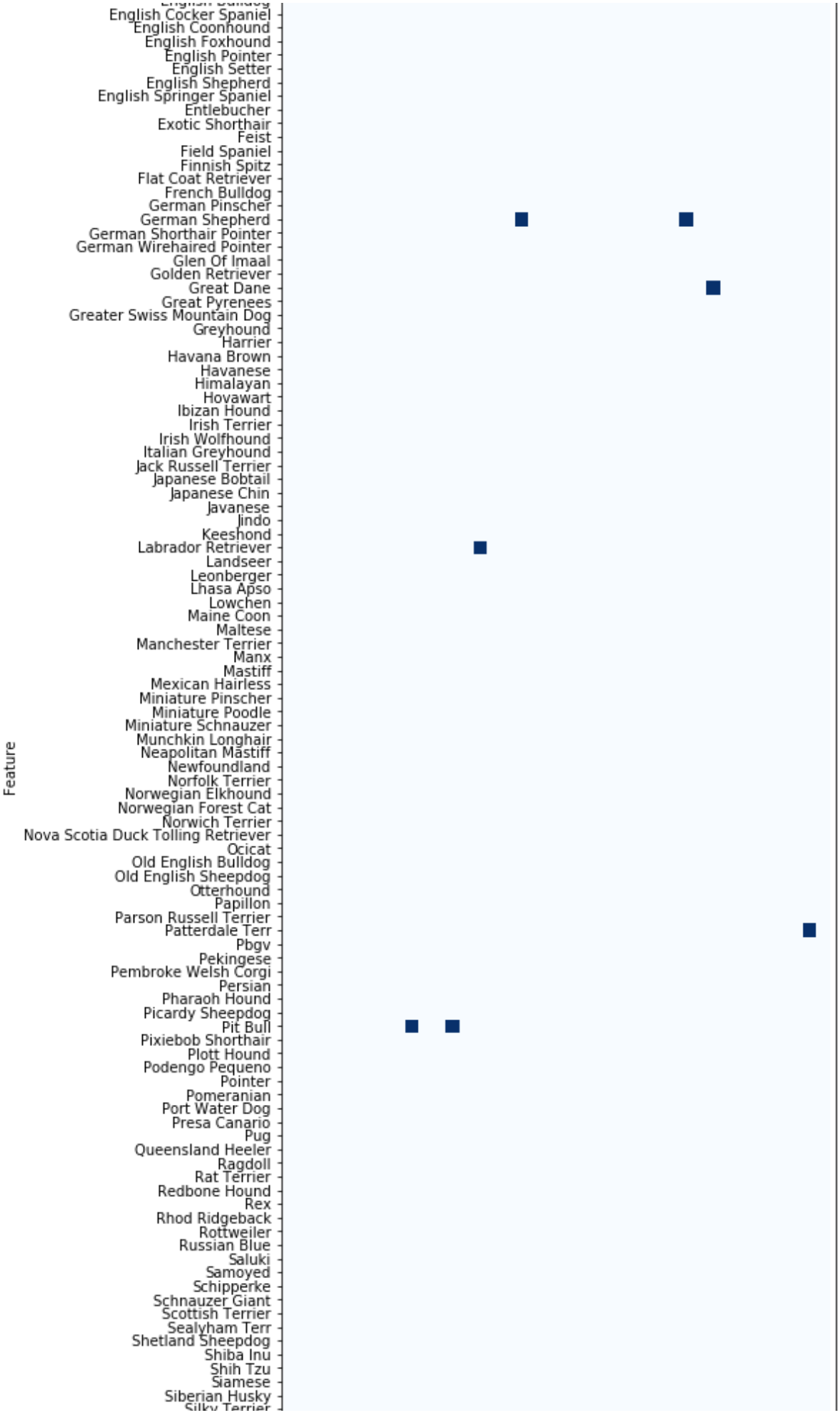
Positive is predicted when the true value is Negative: 988 times.

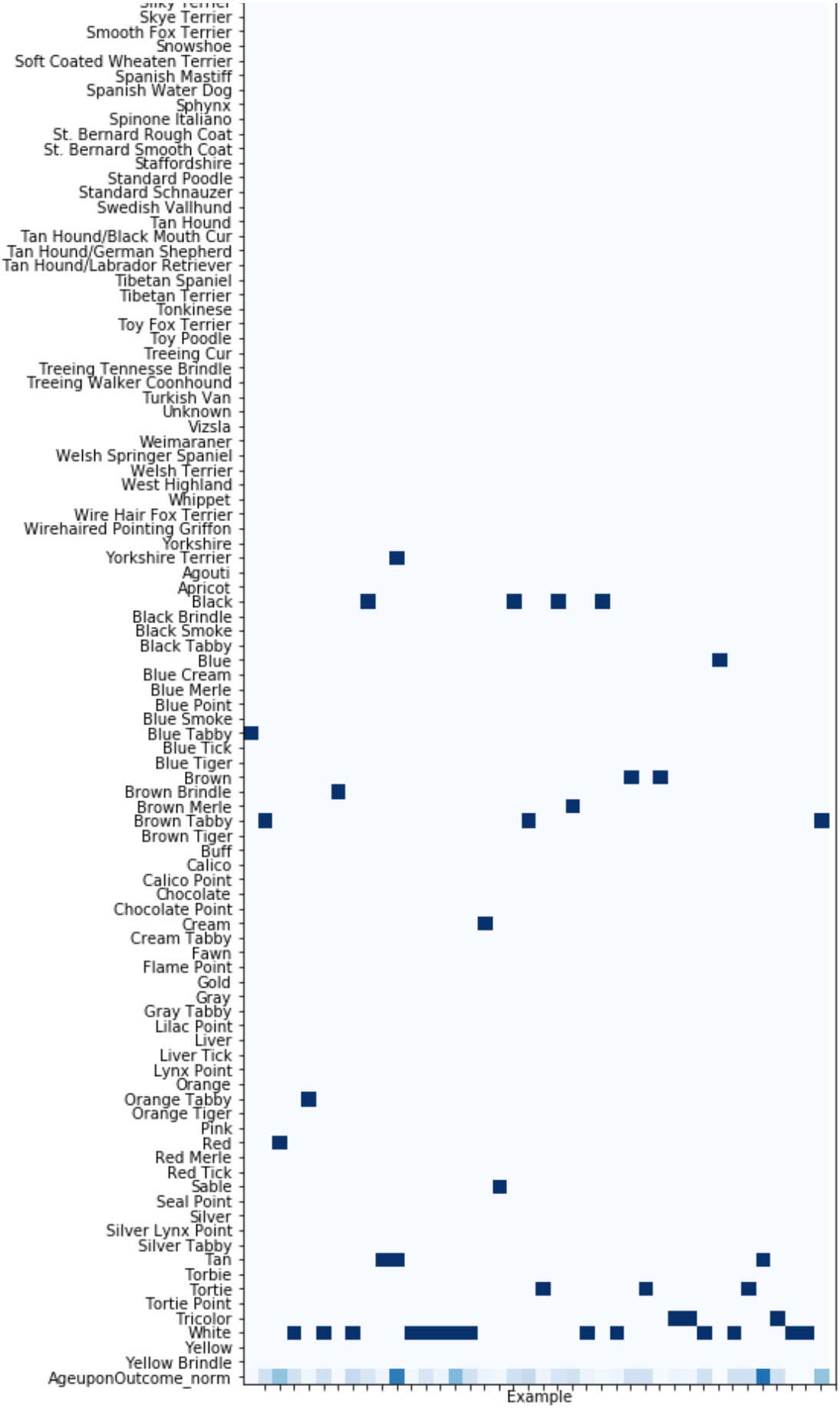

English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier
Feature


Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

Negative is predicted when the true value is Positive: 708 times.




Feature
English Cocker Spaniel
English Coonhound
English Foxhound
English Pointer
English Setter
English Shepherd
English Springer Spaniel
Entlebucher
Exotic Shorthair
Feist
Field Spaniel
Finnish Spitz
Flat Coat Retriever
French Bulldog
German Pinscher
German Shepherd
German Shorthair Pointer
German Wirehaired Pointer
Glen Of Imaal
Golden Retriever
Great Dane
Great Pyrenees
Greater Swiss Mountain Dog
Greyhound
Harrier
Havana Brown
Havanese
Himalayan
Hovawart
Ibizan Hound
Irish Terrier
Irish Wolfhound
Italian Greyhound
Jack Russell Terrier
Japanese Bobtail
Japanese Chin
Javanese
Jindo
Keeshond
Labrador Retriever
Landseer
Leonberger
Lhasa Apso
Lowchen
Maine Coon
Maltese
Manchester Terrier
Manx
Mastiff
Mexican Hairless
Miniature Pinscher
Miniature Poodle
Miniature Schnauzer
Munchkin Longhair
Neapolitan Mastiff
Newfoundland
Norfolk Terrier
Norwegian Elkhound
Norwegian Forest Cat
Norwich Terrier
Nova Scotia Duck Tolling Retriever
Ocicat
Old English Bulldog
Old English Sheepdog
Otterhound
Papillon
Parson Russell Terrier
Patterdale Terr
Pbgv
Pekingese
Pembroke Welsh Corgi
Persian
Pharaoh Hound
Picardy Sheepdog
Pit Bull
Pixiebob Shorthair
Plott Hound
Podengo Pequeno
Pointer
Pomeranian
Port Water Dog
Presa Canario
Pug
Queensland Heeler
Ragdoll
Rat Terrier
Redbone Hound
Rex
Rhod Ridgeback
Rottweiler
Russian Blue
Saluki
Samoyed
Schipperke
Schnauzer Giant
Scottish Terrier
Sealyham Terr
Shetland Sheepdog
Shiba Inu
Shih Tzu
Siamese
Siberian Husky
Silky Terrier


Silky Terrier
Skye Terrier
Smooth Fox Terrier
Snowshoe
Soft Coated Wheaten Terrier
Spanish Mastiff
Spanish Water Dog
Sphynx
Spinone Italiano
St. Bernard Rough Coat
St. Bernard Smooth Coat
Staffordshire
Standard Poodle
Standard Schnauzer
Swedish Vallhund
Tan Hound
Tan Hound/Black Mouth Cur
Tan Hound/German Shepherd
Tan Hound/Labrador Retriever
Tibetan Spaniel
Tibetan Terrier
Tonkinese
Toy Fox Terrier
Toy Poodle
Treeing Cur
Treeing Tennesse Brindle
Treeing Walker Coonhound
Turkish Van
Unknown
Vizsla
Weimaraner
Welsh Springer Spaniel
Welsh Terrier
West Highland
Whippet
Wire Hair Fox Terrier
Wirehaired Pointing Griffon
Yorkshire
Yorkshire Terrier
Agouti
Apricot
Black
Black Brindle
Black Smoke
Black Tabby
Blue
Blue Cream
Blue Merle
Blue Point
Blue Smoke
Blue Tabby
Blue Tick
Blue Tiger
Brown
Brown Brindle
Brown Merle
Brown Tabby
Brown Tiger
Buff
Calico
Calico Point
Chocolate
Chocolate Point
Cream
Cream Tabby
Fawn
Flame Point
Gold
Gray
Gray Tabby
Lilac Point
Liver
Liver Tick
Lynx Point
Orange
Orange Tabby
Orange Tiger
Pink
Red
Red Merle
Red Tick
Sable
Seal Point
Silver
Silver Lynx Point
Silver Tabby
Tan
Torbie
Tortie
Tortie Point
Tricolor
White
Yellow
Yellow Brindle
AgeuponOutcome_norm
Example

If we were to pursue this approach further, we would use this error analysis to guide the feature engineering.

Log Loss functions for the binary outcomes

In [109]:
```
# What is the log loss of the "dumb" classifier that predicts all adoption?

# Create numpy array, pred_prob, that predicts all adoption.
pred_prob = np.zeros((len(train_labels_bin),2))
pred_prob[:, 0] = 1

# Calculate and print log_loss metric for the dumb classifier.
log_loss_metric = log_loss(train_labels_bin, pred_prob
                           , labels=['Negative'
                                     , 'Positive'
                                    ]
                          )
print('Log loss for the dumb classifier is {:.2f}.'.format(log_loss_metric))
```

```
Log loss for the dumb classifier is 20.10.
```

In [110]:
```
# Pick our favorite classifier from above.

indices = np.asarray(range(len(projected_X_train_bin_os)))
X_train_bin, X_test_bin, y_train_bin, y_test_bin,indices_train_bin,indices_tes
t_bin= train_test_split(projected_X_train_bin_os

, train_labels_bin_os

, indices

, test_size = .25

, stratify = train_labels_bin_os

, random_state = 99)


clf_optimal = RandomForestClassifier(max_depth=60, min_samples_split=4, criter
ion='gini')
# Fit it.
clf_optimal.fit(X_train_bin, y_train_bin)
# Predict probabilities.
pred_prob = clf_optimal.predict_proba(X_test_bin)

# Calculate and print log_loss metric.
log_loss_metric = log_loss(y_test_bin, pred_prob
                           , labels=['Negative'
                                     , 'Positive'
                                    ]
                          )
print('Log loss for random forest: {:.2f}'.format(log_loss_metric))
```

```
Log loss for random forest: 1.18
```

In [111]:
```
# Assign the splits again for non-projected data.
indices = np.asarray(range(len(train_data)))
X_train_bin, X_test_bin, y_train_bin, y_test_bin,indices_train_bin,indices_tes
t_bin= train_test_split(train_data

, train_labels_bin

, indices

, test_size = .25

, stratify = train_labels_bin

, random_state = 99)


# Select LogisticRegression classifier.
clf_optimal = LogisticRegression()
# Fit it.
clf_optimal.fit(X_train_bin, y_train_bin)
# Predict probabilities.
pred_prob = clf_optimal.predict_proba(X_test_bin)

# Calculate and print log_loss metric.
log_loss_metric = log_loss(y_test_bin, pred_prob
                           , labels=['Negative'
                                     , 'Positive'
                                    ]
                          )
print('Log loss for logistic regression: {:.2f}'.format(log_loss_metric))
```

```
Log loss for logistic regression: 0.49
```

The error analysis returned a log loss of 1.17 for random forest, a notable improvement over the five-class result, and 0.49 for logistic regression, which is substantially better than our best model using all outcome classes. It makes intuitive sense that we would see better-performing models, as we simplified the outcome classes. However, we would argue that this model, predicting only positive and negative outcomes, would actually be more valuable to shelters. If we were to work with shelters to implement further study of this topic, we would recommend the binary approach.

Bibliography

- “Assessment-of-Storefront-Displays-with-a-Multidisciplinary-Approach-Based-on-Neuromarketing-and-Antropological-Marketing.Pdf.” Accessed November 11, 2018. https://www.researchgate.net/profile/Maurizio_Mauri3/publication/328163427_Assessment_of_Storefront_Disp of-Storefront-Displays-with-a-Multidisciplinary-Approach-based-on-Neuromarketing-and-Antropological-Marketing.pdf#page=259 (https://www.researchgate.net/profile/Maurizio_Mauri3/publication/328163427_Assessment_of_Storefront_Dis of-Storefront-Displays-with-a-Multidisciplinary-Approach-based-on-Neuromarketing-and-Antropological-Marketing.pdf#page=259).
- Chen, Tianqi and Guestrin, Carlos. “XGBooost: A Scalable Tree Boosting System.” SIGKDD. 2016. https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf (https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf)
- Domke, Justin. “Structured Learning via Logistic Regression.” ArXiv:1407.0754 [Cs, Stat], July 2, 2014. http://arxiv.org/abs/1407.0754 (http://arxiv.org/abs/1407.0754).
- Fernandez-Delgado, Manuel, Eva Cernadas, Senen Barro, and Dinani Amorim. “Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?,” n.d., 49.
- Galar, M., A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. “A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches.” IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42, no. 4 (July 2012): 463–84. https://doi.org/10.1109/TSMCC.2011.2161285 (https://doi.org/10.1109/TSMCC.2011.2161285).
- Gultepe, Eren, Jeffrey P. Green, Hien Nguyen, Jason Adams, Timothy Albertson, and Ilias Tagkopoulos. “From Vital Signs to Clinical Outcomes for Patients with Sepsis: A Machine Learning Basis for a Clinical Decision Support System.” Journal of the American Medical Informatics Association: JAMIA 21, no. 2 (April 2014): 315–25. https://doi.org/10.1136/amiajnl-2013-001815 (https://doi.org/10.1136/amiajnl-2013-001815).
- Hawes, Sloane, Josephine Kerrigan, and Kevin Morris. “Factors Informing Outcomes for Older Cats and Dogs in Animal Shelters.” Animals 8, no. 3 (March 7, 2018): 36. https://doi.org/10.3390/ani8030036 (https://doi.org/10.3390/ani8030036).
- Liaw, Andy, and Matthew Wiener. “Classification and Regression by RandomForest.” Forest 23 (November 30, 2001).
- Makridakis, Spyros, Evangelos Spiliotis, and Vassilios Assimakopoulos. “Statistical and Machine Learning Forecasting Methods: Concerns and Ways Forward.” PLOS ONE 13, no. 3 (March 27, 2018): e0194889. https://doi.org/10.1371/journal.pone.0194889 (https://doi.org/10.1371/journal.pone.0194889).
- Pavlyshenko, B. “Machine Learning, Linear and Bayesian Models for Logistic Regression in Failure Detection Problems.” In 2016 IEEE International Conference on Big Data (Big Data), 2046–50, 2016. https://doi.org/10.1109/BigData.2016.7840828 (https://doi.org/10.1109/BigData.2016.7840828).
- “Randomforest2001.Pdf.” Accessed November 11, 2018. https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf (https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf).
- Ruiz, Anne, and Nathalie Villa. “Storms Prediction : Logistic Regression vs Random Forest for Unbalanced Data.” ArXiv:0804.0650 [Math, Stat], April 4, 2008. http://arxiv.org/abs/0804.0650 (http://arxiv.org/abs/0804.0650).
- “Scikit-Learn: Machine Learning in Python — Scikit-Learn 0.20.1 Documentation.” Accessed December 11, 2018. https://scikit-learn.org/stable/ (https://scikit-learn.org/stable/).
- SONG, Yan-yan, and Ying LU. “Decision Tree Methods: Applications for Classification and Prediction.” Shanghai Archives of Psychiatry 27, no. 2 (April 25, 2015): 130–35. https://doi.org/10.11919/j.issn.1002-0829.215044 (https://doi.org/10.11919/j.issn.1002-0829.215044).

- “Tidy Data | Wickham | Journal of Statistical Software.” Accessed November 11, 2018. https://doi.org/10.18637/jss.v059.i10 (https://doi.org/10.18637/jss.v059.i10).