Shelter Cats Exploratory Data Analysis

Mario Tapia-Pacheco

1. Introduction

This dataset comes from the Austin Animal Center, or ACC, based in Austin, Texas. The AAC is the biggest no-kill animal shelter in the United States, sheltering over 11,000 animals on average every year. The data collected dates back to October 1st, 2013 and is updated frequently. I adopted my first pet from a shelter, an adult cat, earlier this year after being told by many others that older cats oftentimes spend more time in shelters than their younger counterparts. This project serves as a tool to investigate whether or not the age of a cat affects its chances of adoption and to search for any other possible factors affecting their adoptability. Learning more about what kinds of cats are not being adopted can help educate the general public and increase the overall adoptability of cats.

The dataset can be found here.

2. Data Preparation and Cleaning

In [1]:
```
# import necessary libraries for data preprocessing
import pandas as pd
import numpy as np
```

In [79]:
```
a_out = pd.read_csv('/Users/mario/Documents/GitHub/aac_cat_adoptability/data/ra
```

In [78]:
```
a_in = pd.read_csv('/Users/mario/Documents/GitHub/aac_cat_adoptability/data/raw
```

In [4]:
```
# merge intake and outcome dataframes into one
data = pd.merge(
    a_in,
    a_out,
    on = ['Animal ID', 'Name', 'Animal Type', 'Breed', 'Color']
)
```

In [5]:
```
# check that data frames were merged correctly
data.head()
```

Out[5]:

	Animal ID	Name	DateTime_x	MonthYear_x	Found Location	Intake Type	Intake Condition	Animal Type	S up Inta
0	A665644	NaN	10/21/2013 07:59:00 AM	October 2013	Austin (TX)	Stray	Sick	Cat	Int Fem
1	A665739	*Alana	10/22/2013 11:11:00 AM	October 2013	Austin (TX)	Stray	Normal	Cat	Int Fem
2	A665763	NaN	10/22/2013 03:10:00 PM	October 2013	E Riverside Dr/Royal Crest Dr in Austin (TX)	Stray	Normal	Dog	Int M
3	A379998	Disciple	10/23/2013 11:42:00 AM	October 2013	51St And Grover in Austin (TX)	Stray	Normal	Dog	Int M
4	A634503	Otter	10/01/2013 02:49:00 PM	October 2013	Manor (TX)	Owner Surrender	Normal	Dog	Spay Fem

In [6]:
```
# inspect dimensions
data.shape
```

Out[6]: (200219, 19)

In [7]:
```
# rename columns to appropriate names
data.rename(columns = {
        'DateTime_x': 'Intake DateTime', 'MonthYear_x': 'Intake MonthYear',
        'DateTime_y': 'Outcome DateTime', 'MonthYear_y': 'Outcome MonthYear'
    }, inplace = True)
```

In [8]:
```
# convert dates and times to pandas datetime objects
data[['Intake DateTime', 'Outcome DateTime']] = data[['Intake DateTime', 'Outco
```

In [9]:
```
# create new variable 'total days in shelter'
data['Total Days in Shelter'] = (data['Outcome DateTime'] - data['Intake DateTi
```

In [80]:
```
data.to_csv('/Users/mario/Documents/GitHub/aac_cat_adoptability/data/preproces
```

Now that we have our files combined into one data frame with important information, we can subset cats as the data has both cats and dogs.

In [10]:
```
cats = data[data['Animal Type'] == 'Cat']
```

In [11]:
```
cats.head()
```

Out[11]:

	Animal ID	Name	Intake DateTime	Intake MonthYear	Found Location	Intake Type	Intake Condition	Animal Type	Sex upon Intake	A u Int
0	A665644	NaN	2013-10-21 07:59:00	October 2013	Austin (TX)	Stray	Sick	Cat	Intact Female	we
1	A665739	*Alana	2013-10-22 11:11:00	October 2013	Austin (TX)	Stray	Normal	Cat	Intact Female	mo
5	A665496	Mikey	2013-10-18 18:07:00	October 2013	12001 Metric Blvd in Austin (TX)	Stray	Normal	Cat	Neutered Male	ye
7	A664936	*Jester	2013-10-11 11:20:00	October 2013	501 U.S. 183 in Austin (TX)	Stray	Normal	Cat	Intact Male	mo
8	A664235	NaN	2013-10-01 08:33:00	October 2013	Abia in Austin (TX)	Stray	Normal	Cat	Unknown	w

In [12]:
```
cats.describe()
```

Out[12]:

	Intake DateTime	Outcome DateTime	Total Days in Shelter
count	66487	66487	66487.000000
mean	2018-05-30 17:22:19.098169600	2018-06-22 07:01:52.761592576	22.569140
min	2013-10-01 08:33:00	2013-10-01 10:39:00	-3347.283333
25%	2015-11-21 15:11:00	2015-12-20 19:05:00	1.175694
50%	2018-04-22 13:28:00	2018-05-11 00:00:00	7.040972
75%	2020-08-31 14:00:30	2020-10-07 13:07:30	33.039236
max	2023-10-17 14:55:00	2023-10-18 11:33:00	3445.252778
std	NaN	NaN	191.265254

In [13]:
```
# in comparison to dogs?
dogs = data[data['Animal Type'] == 'Dog']
dogs.describe()
```

Out[13]:

	Intake DateTime	Outcome DateTime	Total Days in Shelter
count	124646	124646	124646.000000
mean	2018-02-07 02:55:25.066508544	2018-02-26 04:07:03.929849600	19.049755
min	2013-10-01 07:51:00	2013-10-01 11:42:00	-3582.003472
25%	2015-11-29 11:07:00	2015-12-16 17:40:00	0.954167
50%	2017-12-12 14:18:00	2017-12-30 13:08:30	5.331250
75%	2019-11-16 18:05:45	2019-12-05 10:53:00	25.703299
max	2023-10-18 11:00:00	2023-10-18 11:07:00	3592.795139
std	NaN	NaN	335.517215

In [14]:
```
# inspect types of outcomes
cats['Outcome Type'].unique()
```

Out[14]:
```
array(['Transfer', 'Adoption', 'Euthanasia', 'Died', 'Return to Owner',
       'Missing', 'Rto-Adopt', 'Disposal', 'Relocate', nan], dtype=object)
```

We can further subset the data by only choosing observations where cats were adopted. After all, the purpose of this project is to analyze what is affecting adoption rates among cats.

In [15]:
```
cat_adopt = cats[(cats['Outcome Type'] == 'Rto-Adopt') | (cats['Outcome Type']
cat_adopt.head()
```

Out[15]:

	Animal ID	Name	Intake DateTime	Intake MonthYear	Found Location	Intake Type	Intake Condition	Animal Type	Sex upon Intake
1	A665739	*Alana	2013-10-22 11:11:00	October 2013	Austin (TX)	Stray	Normal	Cat	Intact Female
5	A665496	Mikey	2013-10-18 18:07:00	October 2013	12001 Metric Blvd in Austin (TX)	Stray	Normal	Cat	Neutered Male
7	A664936	*Jester	2013-10-11 11:20:00	October 2013	501 U.S. 183 in Austin (TX)	Stray	Normal	Cat	Intact Male
12	A664887	*Gia	2013-10-10 13:48:00	October 2013	1901 Onion Creek Pkwy in Austin (TX)	Stray	Normal	Cat	Intact Female
29	A665398	Haven	2013-10-17 12:26:00	October 2013	Austin (TX)	Owner Surrender	Normal	Cat	Intact Female

In [16]:
```
cat_adopt.describe()
```

Out[16]:

	Intake DateTime	Outcome DateTime	Total Days in Shelter
count	37630	37630	37630.000000
mean	2018-10-13 22:10:48.025511680	2018-11-16 00:33:24.499069696	33.099033
min	2013-10-01 11:51:00	2013-10-01 12:45:00	-3253.647917
25%	2016-06-06 14:25:00	2016-07-09 18:27:45	5.218229
50%	2018-08-31 16:07:00	2018-10-06 19:02:30	20.298264
75%	2021-06-07 11:42:00	2021-07-10 07:48:30	51.872222
max	2023-10-17 14:55:00	2023-10-18 11:33:00	3445.252778
std	NaN	NaN	222.563510

In [17]:
```
cat_adopt.isnull().mean()
```

Out[17]:
```
Animal ID              0.000000
Name                   0.150678
Intake DateTime        0.000000
Intake MonthYear       0.000000
Found Location         0.000000
Intake Type            0.000000
Intake Condition       0.000000
Animal Type            0.000000
Sex upon Intake        0.000000
Age upon Intake        0.000000
Breed                  0.000000
Color                  0.000000
Outcome DateTime       0.000000
Outcome MonthYear      0.000000
Date of Birth          0.000000
Outcome Type           0.000000
Outcome Subtype        0.751422
Sex upon Outcome       0.000000
Age upon Outcome       0.000000
Total Days in Shelter  0.000000
dtype: float64
```

In [18]:
```
cat_adopt['Outcome Subtype'].unique()
```

Out[18]:
```
array(['Foster', nan, 'Offsite', 'Customer S', 'Field', 'Barn', 'Prc'],
      dtype=object)
```

A large proportion of data for the outcome subtype variable is missing, I will choose to remove it since even for the cats that do have that information, there is likely not much more insights that can be found.

In [19]:
```
cat_adopt = cat_adopt.drop(['Outcome Subtype'], axis=1)
cat_adopt.head()
```

Out[19]:

	Animal ID	Name	Intake DateTime	Intake MonthYear	Found Location	Intake Type	Intake Condition	Animal Type	Sex upon Intake
1	A665739	*Alana	2013-10-22 11:11:00	October 2013	Austin (TX)	Stray	Normal	Cat	Intact Female
5	A665496	Mikey	2013-10-18 18:07:00	October 2013	12001 Metric Blvd in Austin (TX)	Stray	Normal	Cat	Neutered Male
7	A664936	*Jester	2013-10-11 11:20:00	October 2013	501 U.S. 183 in Austin (TX)	Stray	Normal	Cat	Intact Male
12	A664887	*Gia	2013-10-10 13:48:00	October 2013	1901 Onion Creek Pkwy in Austin (TX)	Stray	Normal	Cat	Intact Female
29	A665398	Haven	2013-10-17 12:26:00	October 2013	Austin (TX)	Owner Surrender	Normal	Cat	Intact Female

3. Exploratory Analysis and Visualizations

In [29]:
```
import altair as alt
alt.data_transformers.disable_max_rows()
```

Out[29]:
```
DataTransformerRegistry.enable('default')
```

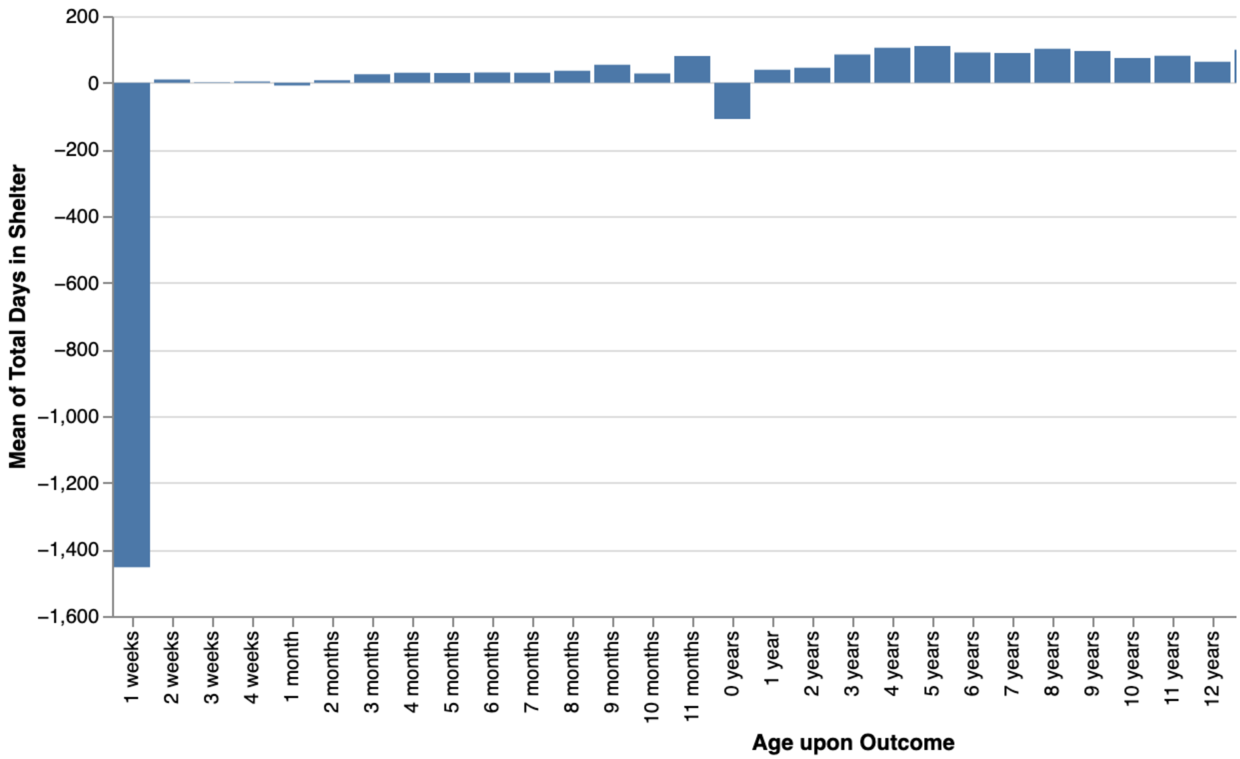
In [30]:
```
cat_adopt['Age upon Outcome'].unique()
```

Out[30]:
```
array(['3 months', '3 years', '4 months', '2 months', '6 months',
       '5 months', '2 years', '7 months', '1 year', '8 years',
       '10 months', '4 years', '7 years', '10 years', '13 years',
       '8 months', '5 years', '6 years', '14 years', '1 month',
       '17 years', '11 months', '16 years', '9 months', '15 years',
       '9 years', '11 years', '12 years', '2 weeks', '20 years',
       '4 weeks', '19 years', '22 years', '3 weeks', '0 years',
       '18 years', '1 weeks'], dtype=object)
```

In [31]:
```
# create sorted ages list
ages = ['1 weeks', '2 weeks', '3 weeks', '4 weeks', '1 month', '2 months',
        '3 months', '4 months', '5 months', '6 months', '7 months', '8 months'
        '9 months', '10 months', '11 months', '0 years', '1 year', '2 years',
        '3 years', '4 years', '5 years', '6 years', '7 years', '8 years',
        '9 years', '10 years', '11 years', '12 years', '13 years', '14 years',
        '15 years', '16 years', '17 years', '18 years', '19 years', '20 years'
        '21 years', '22 years']

# visualize avg days in shelter by age
alt.Chart(cat_adopt).mark_bar().encode(
```

```
    x = alt.X('Age upon Outcome',
              sort = ages),
    y = alt.Y('mean(Total Days in Shelter)')
)
```

Out[31]:



In [32]:
```
# how is 1 week such a negative value? 0 years as well
```

In [33]:
```
cat_adopt[(cat_adopt['Age upon Outcome'] == '1 weeks') | (cat_adopt['Age upon (
```

Out[33]:

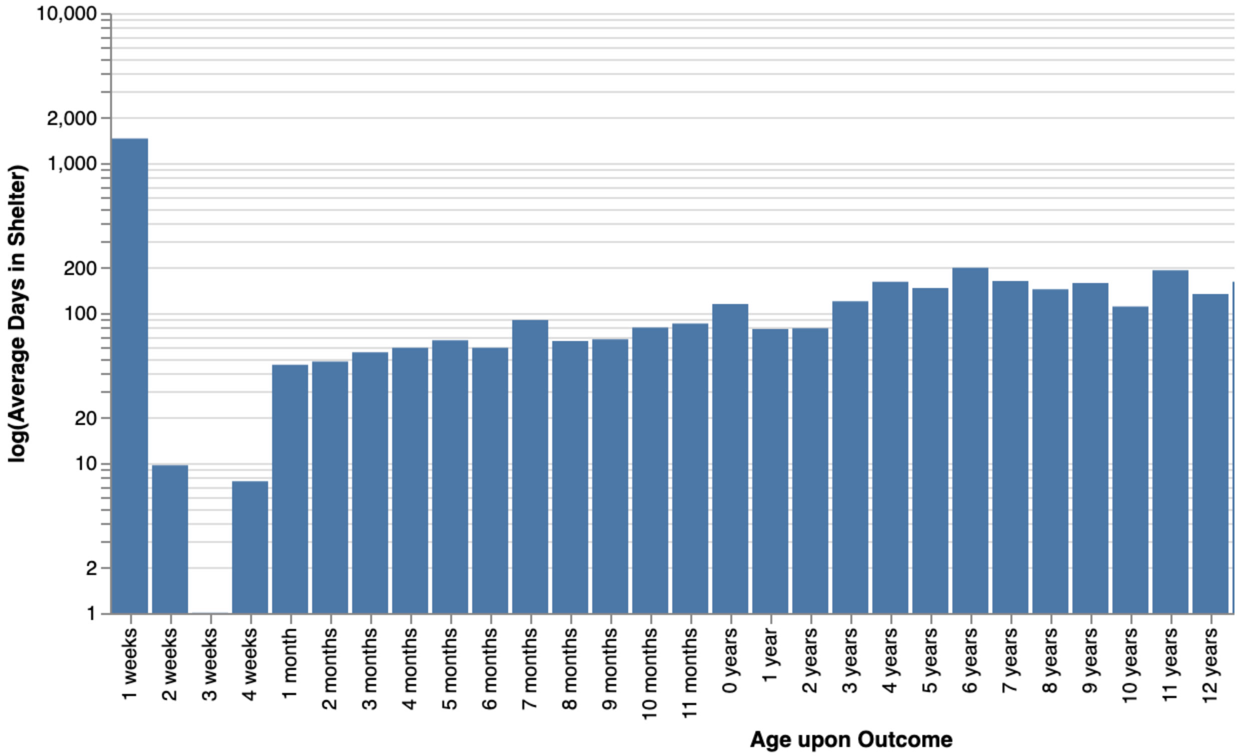
	Animal ID	Name	Intake DateTime	Intake MonthYear	Found Location	Intake Type	Intake Condition	Animal Type	S
66329	A737397	Jellybean	2016-10-27 10:18:00	October 2016	7Th St And Pleasant Valley Rd in Austin (TX)	Stray	Normal	Cat	
66331	A737397	Jellybean	2017-11-15 11:07:00	November 2017	Austin (TX)	Owner Surrender	Normal	Cat	
82503	A853991	NaN	2022-03-28 15:02:00	March 2022	14011 Fm 969 in Austin (TX)	Abandoned	Normal	Cat	N
82505	A853991	NaN	2021-06-07 13:06:00	June 2021	8008 Marble Ridge Drive in Austin (TX)	Stray	Normal	Cat	N
95964	A769632	NaN	2018-04-07 16:59:00	April 2018	1156 W Cesar Chavez in Austin (TX)	Public Assist	Nursing	Cat	U

In [34]:
```
# clearly there has been an error either in my data processing or in the data 
```

In [35]:
```
# make negative observations positive (same magnitude)
cat_adopt[['Total Days in Shelter']] = cat_adopt[['Total Days in Shelter']].ap
```

In [36]:
```
# create sorted ages list
ages = ['1 weeks', '2 weeks', '3 weeks', '4 weeks', '1 month', '2 months',
        '3 months', '4 months', '5 months', '6 months', '7 months', '8 months'
        '9 months', '10 months', '11 months', '0 years', '1 year', '2 years',
        '3 years', '4 years', '5 years', '6 years', '7 years', '8 years',
        '9 years', '10 years', '11 years', '12 years', '13 years', '14 years',
        '15 years', '16 years', '17 years', '18 years', '19 years', '20 years'
        '21 years', '22 years']

# visualize avg days in shelter by age
alt.Chart(cat_adopt).mark_bar().encode(
    x = alt.X('Age upon Outcome',
              sort = ages),
    y = alt.Y('mean(Total Days in Shelter)',
              scale=alt.Scale(type='log', zero=False),
              title = 'log(Average Days in Shelter)')
)
```

Out[36]:



Besides within the first month a kitten is born, average days in shelter seems to be more or less evenly distributed among different age groups.

In [37]:
```
# create new variable 'age upon outcome in days'
cat_adopt['Age upon Outcome in Days'] = 0
```

In [38]:
```
# input correct values into column
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '1 weeks', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '2 weeks', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '3 weeks', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '4 weeks', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '1 month', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '2 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '3 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '4 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '5 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '6 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '7 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '8 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '9 months', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '10 months', 'Age upon Outcome
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '11 months', 'Age upon Outcome
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '0 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '1 year', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '2 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '3 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '4 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '5 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '6 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '7 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '8 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '9 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '10 years', 'Age upon Outcome i
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '11 years', 'Age upon Outcome i
```

```
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '12 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '13 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '14 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '15 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '16 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '17 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '18 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '19 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '20 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '21 years', 'Age upon Outcome in
cat_adopt.loc[cat_adopt['Age upon Outcome'] == '22 years', 'Age upon Outcome in

cat_adopt[['Age upon Outcome in Days', 'Age upon Outcome', 'Total Days in Shel
```

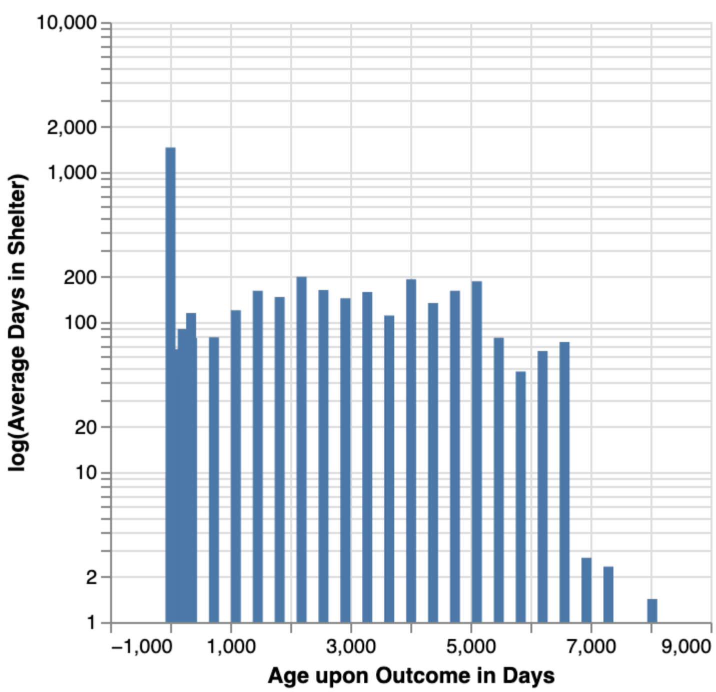
Out[38]:

	Age upon Outcome in Days	Age upon Outcome	Total Days in Shelter
1	90	3 months	59.267361
5	1095	3 years	3.990972
7	120	4 months	68.290278
12	90	3 months	31.130556
29	60	2 months	24.179861

In [39]:

```
alt.Chart(cat_adopt).mark_bar().encode(
    x = alt.X('Age upon Outcome in Days',
              sort = ages),
    y = alt.Y('mean(Total Days in Shelter)',
              scale=alt.Scale(type='log', zero=False),
              title = 'log(Average Days in Shelter)')
)
```

Out[39]:

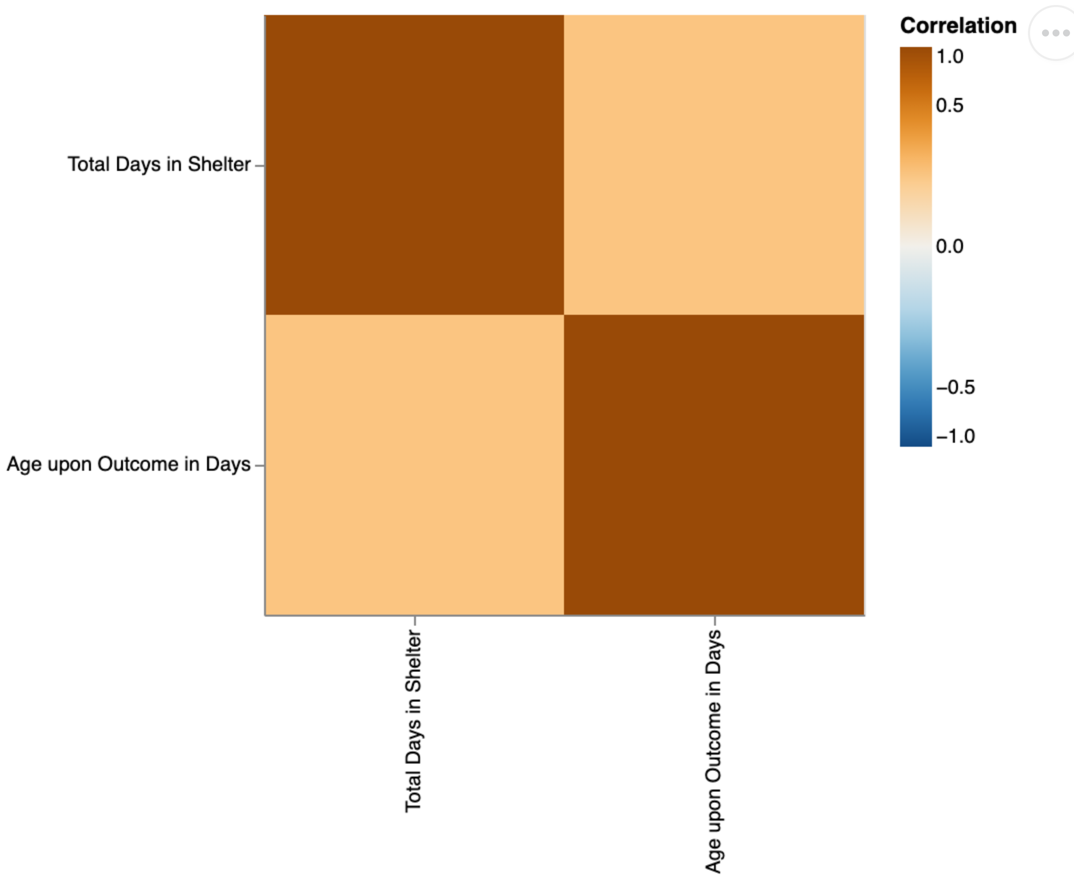


Again, there seems to be a relatively even distribution. Next, we can inspect the correlation heatmap.

In [42]:

```
# create correlation matrix
corr_mx = cat_adopt.loc[:,['Total Days in Shelter', 'Age upon Outcome in Days']

# melt corr_mx
corr_mx_long = corr_mx.reset_index().rename(
    columns = {'index': 'row'}
).melt(
    id_vars = 'row',
    var_name = 'col',
    value_name = 'Correlation'
)

# construct plot
alt.Chart(corr_mx_long).mark_rect().encode(
    x = alt.X('col', title = '', sort = {'field': 'Correlation', 'order': 'asce
    y = alt.Y('row', title = '', sort = {'field': 'Correlation', 'order': 'asce
    color = alt.Color('Correlation',
                      scale = alt.Scale(scheme = 'blueorange', # diverging grad
                                        domain = (-1, 1), # ensure white = 0
                                        type = 'sqrt'), # adjust gradient scale
                     legend = alt.Legend(tickCount = 5)) # add ticks to colorba
).properties(width = 300, height = 300)
```

Out[42]:



In [81]:

```
cat_adopt.to_csv('/Users/mario/Documents/GitHub/aac_cat_adoptability/data/prepr
```

We only have 2 numeric variables to construct a heatmap and they are only slightly positively correlated. This is not very informative, however, we won't have to deal with decorrelating features.

4. Unsupervised Learning (K-Means Clustering)

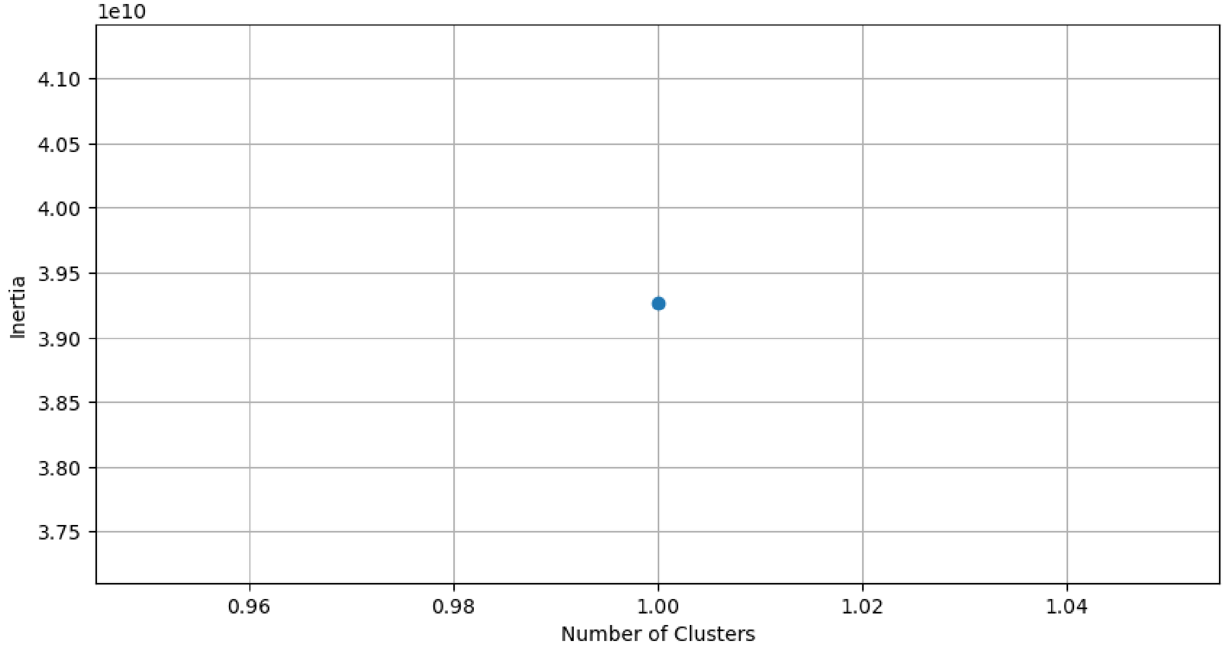
In [43]:
```
# load necessary libraries
import matplotlib.pyplot as plt

from sklearn.cluster import KMeans
```

Although scaling the observations results in more interesting patterns, I will not scale them as the two numeric variables are already using the same units. Furthermore, standardizing them results in negative values which do not make sense for variables measuring time.

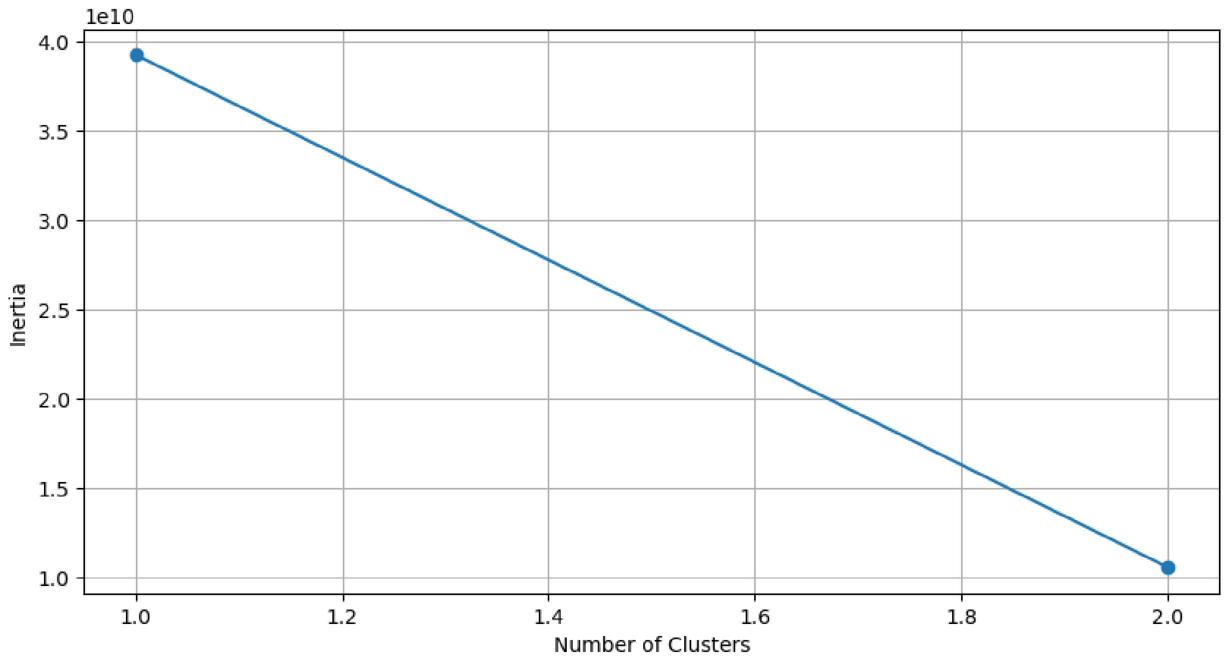
In [44]:
```
# create function for elbow plot of clusters
def optimize_k_means(data, max_k):
  means = []
  inertias = []

  for k in range(1, max_k):
    kmeans = KMeans(n_clusters = k)
    kmeans.fit(data)

    means.append(k)
    inertias.append(kmeans.inertia_)

    # Generate elbow plot
    fig = plt.subplots(figsize = (10, 5))
    plt.plot(means, inertias, 'o-')
    plt.xlabel('Number of Clusters')
    plt.ylabel('Inertia')
    plt.grid(True)
    plt.show()
```

In [45]:
```
# use function and inspect elbow plot
optimize_k_means(cat_adopt[['Age upon Outcome in Days', 'Total Days in Shelter
```

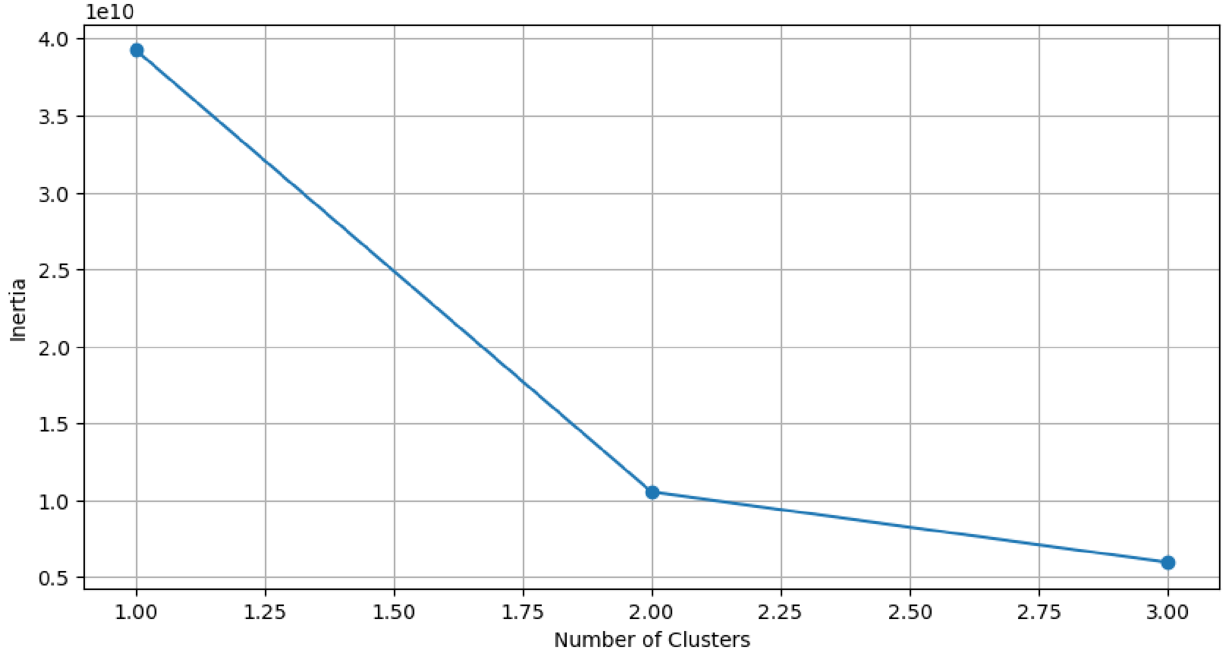
```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



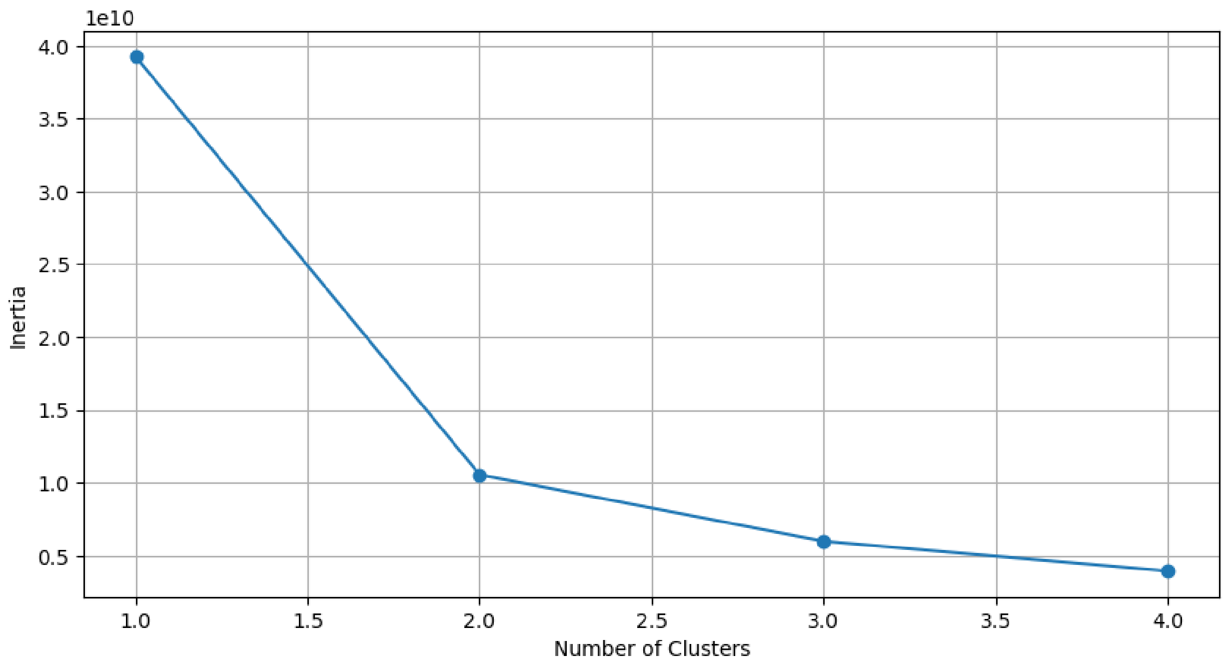

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



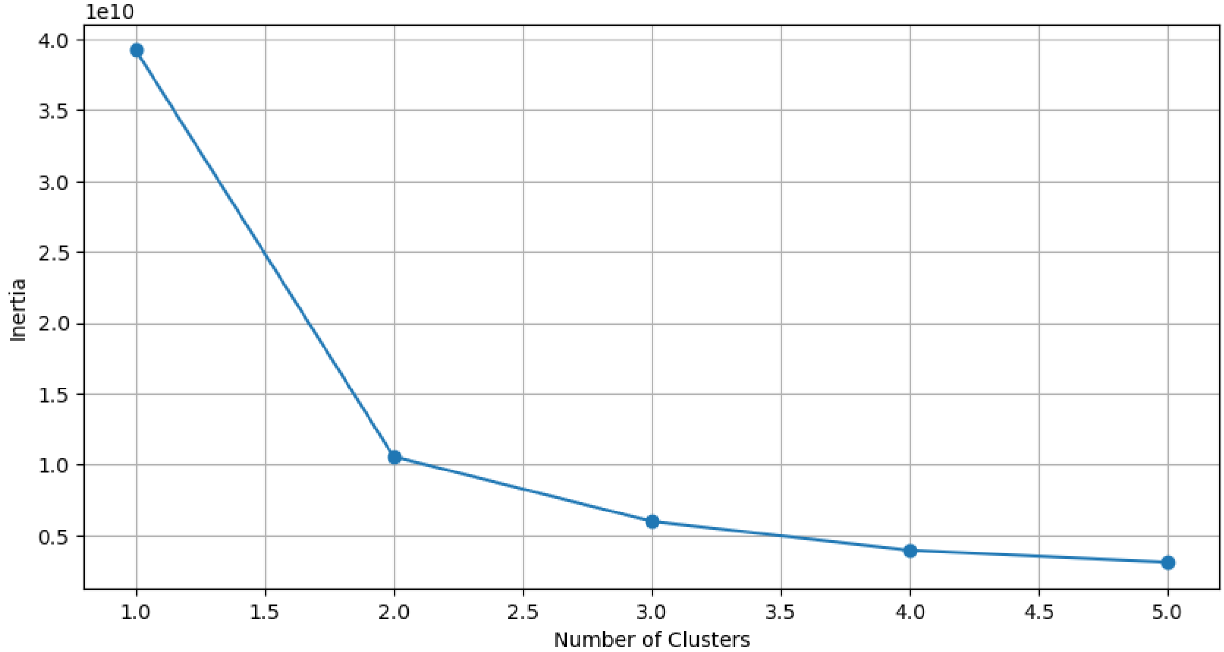

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



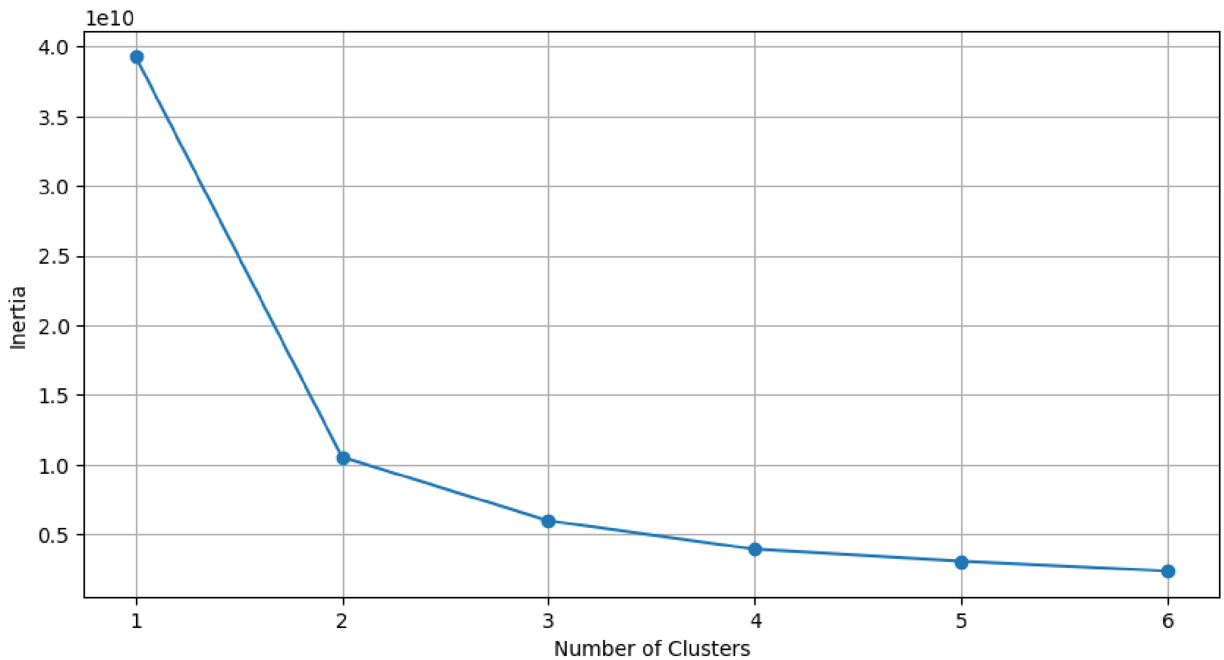

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



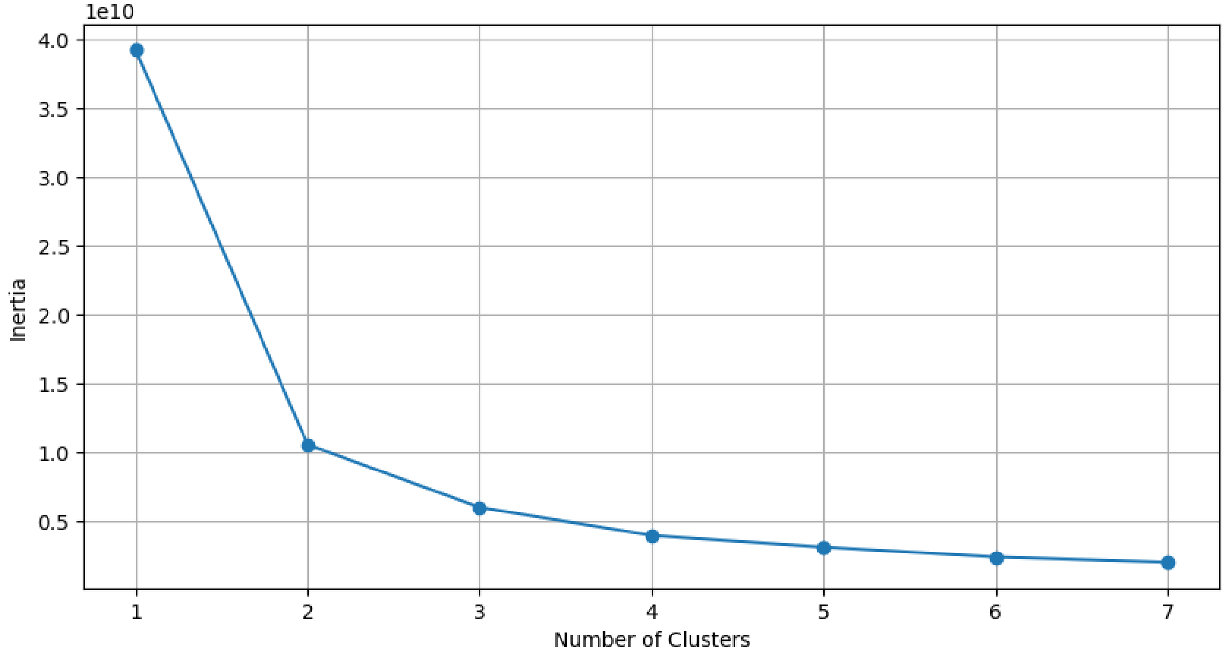

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



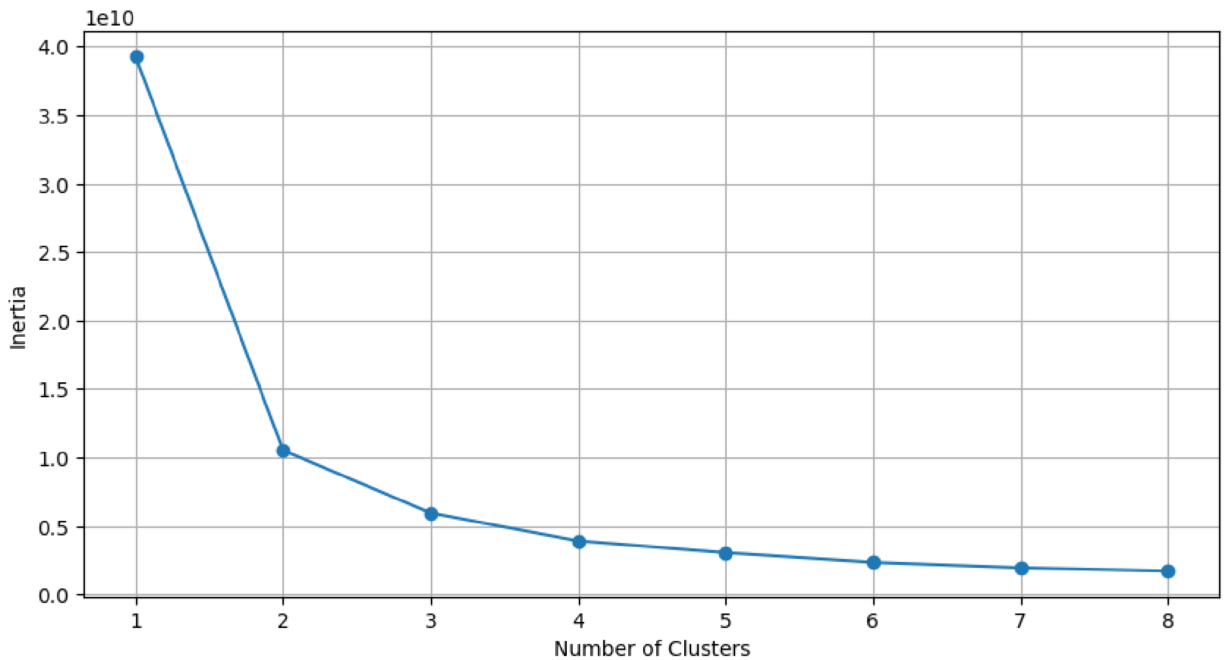

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```




```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```




```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



Based on the elbow plot, I will choose 4 clusters as my k.

In [46]:
```
kmeans = KMeans(n_clusters = 4)
```

In [48]:
```
kmeans.fit(cat_adopt[['Age upon Outcome in Days', 'Total Days in Shelter']])
```

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

Out[48]:

```
▾ KMeans
KMeans(n_clusters=4)
```

In [49]:

```
# assign cluster label to every observation
cat_adopt['kmeans_4'] = kmeans.labels_
```

In [50]:

```
cat_adopt.head()
```

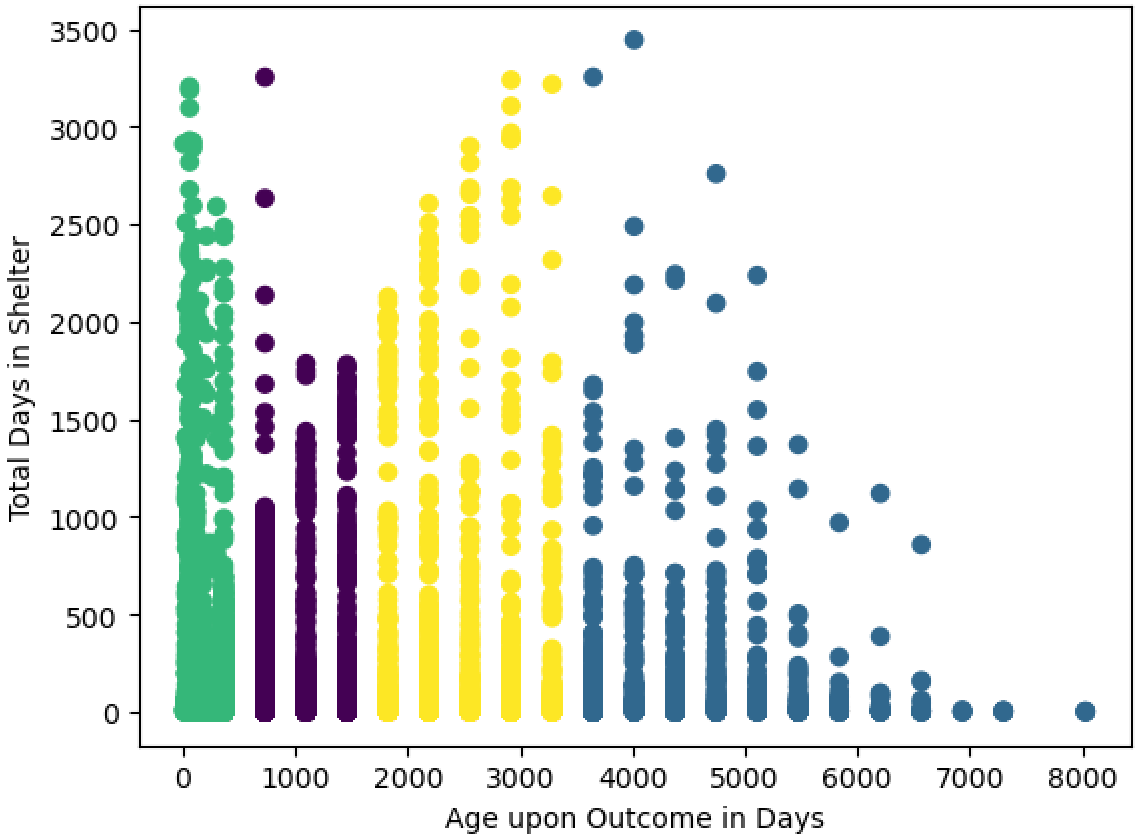
Out[50]:

	Animal ID	Name	Intake DateTime	Intake MonthYear	Found Location	Intake Type	Intake Condition	Animal Type	Sex upon Intake
1	A665739	*Alana	2013-10-22 11:11:00	October 2013	Austin (TX)	Stray	Normal	Cat	Intact Female
5	A665496	Mikey	2013-10-18 18:07:00	October 2013	12001 Metric Blvd in Austin (TX)	Stray	Normal	Cat	Neutered Male
7	A664936	*Jester	2013-10-11 11:20:00	October 2013	501 U.S. 183 in Austin (TX)	Stray	Normal	Cat	Intact Male
12	A664887	*Gia	2013-10-10 13:48:00	October 2013	1901 Onion Creek Pkwy in Austin (TX)	Stray	Normal	Cat	Intact Female
29	A665398	Haven	2013-10-17 12:26:00	October 2013	Austin (TX)	Owner Surrender	Normal	Cat	Intact Female

5 rows × 21 columns

In [51]:

```
# visualize
plt.scatter(x=cat_adopt['Age upon Outcome in Days'], y=cat_adopt['Total Days in
plt.xlabel('Age upon Outcome in Days')
plt.ylabel('Total Days in Shelter')
plt.show()
```

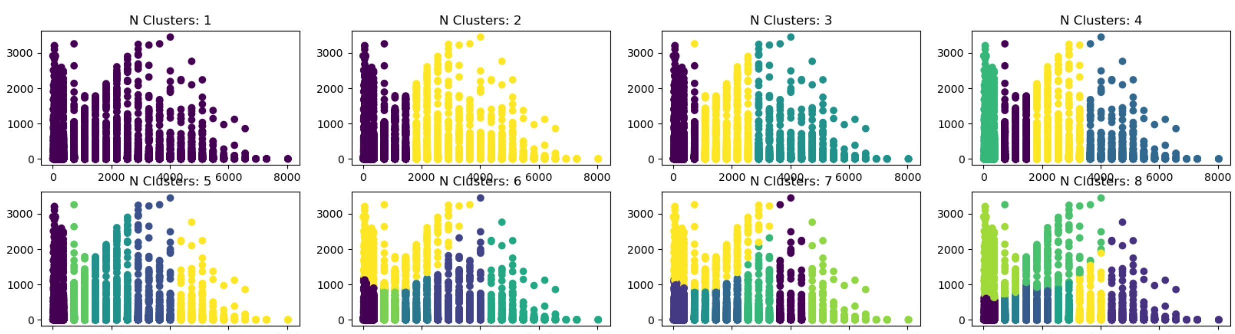


The visualization when observations are not standardized is unfortunately is not very informative as it seems to just split cats into kittens, young adult cats, middle aged cats, and senior cats. Perhaps this is not informative as I had expected these to be the groups of cats, but there is no variation of clusters based on total days in shelter, only by age upon outcome. However, once we standardize there seems to be more interesting clusters. Standardizing here may not make sense since it results in negative ages and time spent in the shelter. When we standardize the two variables we can see that there is a small group of kittens that get adopted quickly, a small group of young adults that get adopted quickly, a surprisingly decent amount of senior cats adopted quickly, cats of all ages adopted somewhat promptly, and cats of all ages who spend a lot more time in the shelter.

In [52]:
```
# add cluster labels for different values of k
for k in range(1,9):
  kmeans = KMeans(n_clusters = k)
  kmeans.fit(cat_adopt[['Total Days in Shelter', 'Age upon Outcome in Days']])
  cat_adopt[f'kmeans_{k}'] = kmeans.labels_
```

```
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Users/mario/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.p
y:1412: FutureWarning: The default value of `n_init` will change from 10 to 'a
uto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

In [53]:
```
# visualize clusters for different values of k
fig, axs = plt.subplots(nrows=2, ncols=4, figsize=(20, 5))

for i, ax in enumerate(fig.axes, start=1):
  ax.scatter(x=cat_adopt['Age upon Outcome in Days'], y=cat_adopt['Total Days i
  ax.set_title(f'N Clusters: {i}')
```



These visualizations do not tell us much, they only divide cats into different age groups. While new patterns emerge when the number of clusters grows larger, we also run into the risk of overfitting the training data.

Model Training

Now that I have explored my data set and familiarized myself with its features, I will build different neural networks in an attempt to predict how many days a cat will spend in the shelter before getting adopted.

In [56]:
```
# import necessary libraries
import random

import tensorflow as tf
from tensorflow import keras

from sklearn.model_selection import train_test_split

import pickle
```

```
2023-12-29 19:57:59.184047: I tensorflow/core/platform/cpu_feature_guard.cc:18
2] This TensorFlow binary is optimized to use available CPU instructions in pe
rformance-critical operations.
To enable the following instructions: AVX2 AVX512F AVX512_VNNI FMA, in other o
perations, rebuild TensorFlow with the appropriate compiler flags.
```

In [57]:
```
# inspect columns
cat_adopt.columns
```

Out[57]:
```
Index(['Animal ID', 'Name', 'Intake DateTime', 'Intake MonthYear',
       'Found Location', 'Intake Type', 'Intake Condition', 'Animal Type',
       'Sex upon Intake', 'Age upon Intake', 'Breed', 'Color',
       'Outcome DateTime', 'Outcome MonthYear', 'Date of Birth',
       'Outcome Type', 'Sex upon Outcome', 'Age upon Outcome',
       'Total Days in Shelter', 'Age upon Outcome in Days', 'kmeans_4',
       'kmeans_1', 'kmeans_2', 'kmeans_3', 'kmeans_5', 'kmeans_6', 'kmeans_7',
       'kmeans_8'],
      dtype='object')
```

In [58]:
```
# create response variable vector and predictor variable matrix
y = cat_adopt['Total Days in Shelter']
X = cat_adopt[['Found Location', 'Intake Type', 'Intake Condition',
               'Sex upon Intake', 'Age upon Intake', 'Breed', 'Color']]
```

In [59]:
```
categorical_columns = ['Found Location', 'Intake Type', 'Intake Condition',
               'Sex upon Intake', 'Age upon Intake', 'Breed', 'Color']

# create dummy variables for categorical columns
dummy_cols = pd.get_dummies(X)

# crop original categorical columns from df
X = X.drop(categorical_columns, axis=1)

# concatenate dummy columns with df
X = pd.concat([X, dummy_cols], axis=1)

# convert dummy columns to float32 for compatability w nn
X = X.astype('float32')

X.head()
```

Out[59]:

	Found Location_- in Austin (TX)	Found Location_0 Highway 290 in Austin (TX)	Found Location_0 S Lampasas St in Travis (TX)	Found Location_0611 Corpus Christi Dr in Austin (TX)	Found Location_10 Olmos Drive in Austin (TX)	Found Location_100 Congress in Austin (TX)	Found Location_10 Elizabeth i Austin (TX
1	0.0	0.0	0.0	0.0	0.0	0.0	0.
5	0.0	0.0	0.0	0.0	0.0	0.0	0.
7	0.0	0.0	0.0	0.0	0.0	0.0	0.
12	0.0	0.0	0.0	0.0	0.0	0.0	0.
29	0.0	0.0	0.0	0.0	0.0	0.0	0.

5 rows × 12194 columns

In [60]:
```
# split data into training and testing data sets
random.seed(122223)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Fitting Feed Forward Neural Networks

In [62]:
```
# configure NN architecture
ff_model = tf.keras.Sequential([
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(units = 64, input_dim = 12194, activation = 'relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1),
    tf.keras.layers.Activation(activation = 'sigmoid')
])
```

In [63]:
```
# configure for training
ff_model.compile(
  loss = 'mean_absolute_error',
  optimizer = 'adam',
  metrics = ['mae']
)
```

In [64]:
```
# fit nn
ff_model.fit(X_train, y_train, validation_split = 0.2, epochs = 5)
```

```
Epoch 1/5
753/753 [==============================] - 22s 27ms/step - loss: 71.7485 - ma
e: 71.7485 - val_loss: 75.0618 - val_mae: 75.0618
Epoch 2/5
753/753 [==============================] - 21s 28ms/step - loss: 71.7208 - ma
e: 71.7208 - val_loss: 75.0617 - val_mae: 75.0617
Epoch 3/5
753/753 [==============================] - 22s 30ms/step - loss: 71.7207 - ma
e: 71.7207 - val_loss: 75.0617 - val_mae: 75.0617
Epoch 4/5
753/753 [==============================] - 25s 34ms/step - loss: 71.7206 - ma
e: 71.7206 - val_loss: 75.0617 - val_mae: 75.0617
Epoch 5/5
753/753 [==============================] - 24s 32ms/step - loss: 71.7207 - ma
e: 71.7207 - val_loss: 75.0617 - val_mae: 75.0617
```

Out[64]:
```
<keras.src.callbacks.History at 0x1432288d0>
```

In [65]:
```
ff_model.evaluate(X_train, y_train)
```

```
941/941 [==============================] - 6s 6ms/step - loss: 72.3889 - mae:
72.3889
```

Out[65]:
```
[72.38890075683594, 72.38890075683594]
```

In [82]:
```
# export model
pickle.dump(ff_model, open('/Users/mario/Documents/GitHub/aac_cat_adoptability/
```

In [66]:
```
# try a different nn architecture
ff_model_2 = tf.keras.Sequential([
    tf.keras.layers.Dense(units = 128, input_dim = X_train.shape[1], activatio
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dense(1, activation = 'linear')
])
```

In [67]:
```
# configure for training
ff_model_2.compile(
  loss = 'mean_absolute_error',
  optimizer = 'adam',
  metrics = ['mae']
)
```

In [68]:
```
# fit nn
ff_model_2.fit(X_train, y_train, validation_split = 0.2, epochs = 5)
```

```
Epoch 1/5
753/753 [==============================] - 31s 40ms/step - loss: 59.1044 - ma
e: 59.1044 - val_loss: 61.8076 - val_mae: 61.8076
Epoch 2/5
753/753 [==============================] - 35s 46ms/step - loss: 55.8499 - ma
e: 55.8499 - val_loss: 61.0917 - val_mae: 61.0917
Epoch 3/5
753/753 [==============================] - 33s 44ms/step - loss: 53.8912 - ma
e: 53.8912 - val_loss: 60.5458 - val_mae: 60.5458
Epoch 4/5
753/753 [==============================] - 35s 46ms/step - loss: 52.2462 - ma
e: 52.2462 - val_loss: 61.1018 - val_mae: 61.1018
Epoch 5/5
753/753 [==============================] - 35s 47ms/step - loss: 50.8511 - ma
e: 50.8511 - val_loss: 60.9676 - val_mae: 60.9676
```

Out[68]:
```
<keras.src.callbacks.History at 0x145fc6410>
```

In [69]:
```
ff_model_2.evaluate(X_train, y_train)
```

```
941/941 [==============================] - 11s 11ms/step - loss: 51.6508 - ma
e: 51.6508
```

Out[69]:
```
[51.650760650634766, 51.650760650634766]
```

In [83]:
```
# export model
pickle.dump(ff_model_2, open('/Users/mario/Documents/GitHub/aac_cat_adoptabili
```

In [70]:
```
# configure yet another NN architecture
ff_model_3 = tf.keras.Sequential([
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(units = 128, input_dim = X_train.shape[1], activatio
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(units = 256, activation = 'relu'),
    tf.keras.layers.Dense(1, activation = 'linear')
])
```

In [71]:
```
# configure for training
ff_model_3.compile(
  loss = 'mean_absolute_error',
  optimizer = 'adam',
  metrics = ['mae']
)
```

In [72]:
```
# fit nn
ff_model_3.fit(X_train, y_train, validation_split = 0.2, epochs = 5)
```

```
Epoch 1/5
753/753 [==============================] - 39s 50ms/step - loss: 60.3213 - ma
e: 60.3213 - val_loss: 62.8589 - val_mae: 62.8589
Epoch 2/5
753/753 [==============================] - 37s 49ms/step - loss: 57.9800 - ma
e: 57.9800 - val_loss: 61.0716 - val_mae: 61.0716
Epoch 3/5
753/753 [==============================] - 47s 62ms/step - loss: 56.6473 - ma
e: 56.6473 - val_loss: 61.2151 - val_mae: 61.2151
Epoch 4/5
753/753 [==============================] - 41s 55ms/step - loss: 55.5834 - ma
e: 55.5834 - val_loss: 61.2144 - val_mae: 61.2144
Epoch 5/5
753/753 [==============================] - 42s 56ms/step - loss: 54.7618 - ma
e: 54.7618 - val_loss: 60.9936 - val_mae: 60.9936
```

Out[72]:
```
<keras.src.callbacks.History at 0x1461deed0>
```

In [73]:
```
ff_model_3.evaluate(X_train, y_train)
```

```
941/941 [==============================] - 9s 9ms/step - loss: 53.9523 - mae:
53.9523
```

Out[73]:
```
[53.952274322509766, 53.952274322509766]
```

In [84]:
```
# export model
pickle.dump(ff_model_3, open('/Users/mario/Documents/GitHub/aac_cat_adoptabili
```

After building three neural networks, we can see that performance is highly dependent on our choices when constructing the architecture for the network. The activation functions, layers, and amount of nodes in each layer not only affected model performance, but also runtime. A deeper model may perform better, but would also require more resources.

Now we must investigate other kinds of models that can solve this problem as neural networks are not always the best models for a problem. How does a random forest perform on this data set?

Fitting a Random Forest

In [77]:
```
# import necessary libraries
import sklearn

from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score

from sklearn.model_selection import cross_validate
from prettytable import PrettyTable
```

In [85]:
```
hyperparameter_score_list = []

# max_features, mtry
mtry = range(1,8)
# n_estimators, trees
trees = [5, 10]
```

```
# min_samples_split, min_n
min_n = [3, 4000, 9000, 14000, 20000]

# store 5 fold cross validation results
for m in mtry:
    for t in trees:
      for n in min_n:
        rf = RandomForestRegressor(max_features=m,
                                   n_estimators=t, min_samples_split=n)
        scores = cross_validate(rf, X_train, y_train,
                                cv=5, scoring='neg_mean_absolute_error')
        mean_score = np.mean(scores['test_score'])
        hyperparameter_score_list.append([m, t, n, mean_score])
```

In [86]:

```
# display cross validation scores
score_table = PrettyTable(["mtry", "n_trees", "min_node_size", "avg mae"])
for row in hyperparameter_score_list:
    score_table.add_row([row[0], row[1], row[2], round(row[3],4)])
print(score_table)
```

mtry	n_trees	min_node_size	avg mae
1	5	3	-87.3599
1	5	4000	-81.4764
1	5	9000	-81.5964
1	5	14000	-82.2243
1	5	20000	-82.4981
1	10	3	-85.539
1	10	4000	-81.3884
1	10	9000	-81.5919
1	10	14000	-82.0537
1	10	20000	-82.4955
2	5	3	-87.507
2	5	4000	-81.6895
2	5	9000	-81.6323
2	5	14000	-82.1904
2	5	20000	-82.6536
2	10	3	-85.2736
2	10	4000	-81.2787
2	10	9000	-81.6863
2	10	14000	-82.1502
2	10	20000	-82.2846
3	5	3	-86.9668
3	5	4000	-81.8139
3	5	9000	-81.7341
3	5	14000	-82.2261
3	5	20000	-82.4871
3	10	3	-85.8149
3	10	4000	-80.9313
3	10	9000	-81.7463
3	10	14000	-82.0585
3	10	20000	-82.4318
4	5	3	-86.0751
4	5	4000	-81.0582
4	5	9000	-81.5654
4	5	14000	-82.2206
4	5	20000	-82.6232
4	10	3	-85.0665
4	10	4000	-80.8748
4	10	9000	-81.4571
4	10	14000	-82.1326
4	10	20000	-82.5525
5	5	3	-86.4376
5	5	4000	-81.2041
5	5	9000	-81.7786
5	5	14000	-82.0753
5	5	20000	-82.3082
5	10	3	-85.1906
5	10	4000	-80.9205
5	10	9000	-81.287
5	10	14000	-82.1644
5	10	20000	-82.4841
6	5	3	-86.658
6	5	4000	-80.995
6	5	9000	-81.6789
6	5	14000	-82.2836
6	5	20000	-82.5484
6	10	3	-85.0088
6	10	4000	-81.0474

```
|  6   |    10    |      9000      | -81.096  |
|  6   |    10    |     14000      | -81.9478 |
|  6   |    10    |     20000      | -82.6286 |
|  7   |    5     |       3        | -86.5533 |
|  7   |    5     |      4000      | -81.5138 |
|  7   |    5     |      9000      | -81.5704 |
|  7   |    5     |     14000      | -81.955  |
|  7   |    5     |     20000      | -82.3519 |
|  7   |    10    |       3        | -84.9485 |
|  7   |    10    |      4000      | -80.9226 |
|  7   |    10    |      9000      | -81.4924 |
|  7   |    10    |     14000      | -81.9582 |
|  7   |    10    |     20000      | -82.5344 |
+------+----------+----------------+----------+
```

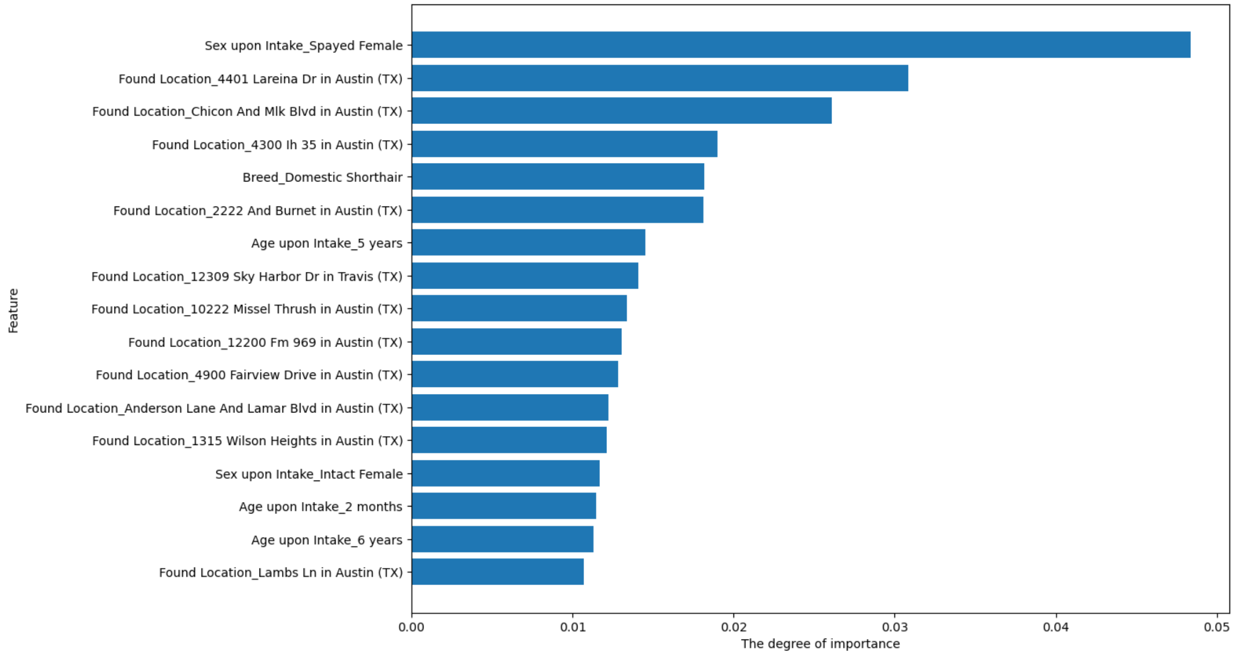
By inspecting the results table, it is apparent that the random forest with max_features = 4, n_estimators = 10, and min_samples_split = 4000 performs the best on the training data. Thus, we will choose those parameters for our random forest model.

In [87]:
```
# create rf with chosen parameters
rf = RandomForestRegressor(max_features = 4,
                                    n_estimators = 10, min_samples_split = 4000
best_rf = rf.fit(X_train, y_train)
```

In [88]:
```
# export model
pickle.dump(best_rf, open('/Users/mario/Documents/GitHub/aac_cat_adoptability/
```

In [90]:
```
# load model
loaded_rf = pickle.load(open('/Users/mario/Documents/GitHub/aac_cat_adoptabili
```

In [91]:
```
# view variable importance
features_mask = loaded_rf.feature_importances_ > 0.01

# filter and get indices of sorted importances > 0.01
sorted_indices = (loaded_rf.feature_importances_[features_mask]).argsort()[::-

# sort importances and names
sorted_importances = loaded_rf.feature_importances_[features_mask][sorted_indi
sorted_features = X_train.columns[features_mask][sorted_indices]

# set plot window
fig, ax = plt.subplots(figsize=(12, 9))

# plot
ax.barh(range(len(sorted_features)), sorted_importances, align='center')
ax.set_yticks(range(len(sorted_features)))
ax.set_yticklabels(sorted_features)
ax.set_xlabel("The degree of importance")
ax.set_ylabel("Feature")
ax.invert_yaxis()  # invert y-axis to display most important feature at the to

plt.show()
```



I did not expect the found location to play a significant role in predicting time spelt in the shelter for cats.

In [92]:
```
mean_absolute_error(y_train, loaded_rf.predict(X_train))
```

Out[92]:
```
77.93671249847293
```

77.94 is greater than all the mean absolute error values for our neural networks, thus the random forest performed worse than the feed forward neural networks. I will now evaluate the best performing model on the training data.

In [93]:
```
ff_model_2.evaluate(X_test, y_test)
```

```
236/236 [==============================] - 2s 9ms/step - loss: 57.6828 - mae:
57.6828
```

Out[93]:
```
[57.6828498840332, 57.6828498840332]
```

In [99]:
```
r2_score(y_test, ff_model_2.predict(X_test))
```

```
236/236 [==============================] - 1s 6ms/step
```

Out[99]:
```
-0.03710700288709323
```

The second neural network performed poorly on the data. However, it performed better than the other neural network and random forest model that we fit. This data set is not very optimal for machine learning as it had no numerical data, instead we had to create two variables for age and time spent in the shelter. Regardless, the neural network outperformed the other two models on the training data.

5. Summary and Conclusion

Through the testing and experimentation of a handul of models, we were able to find that the best predictive model for predicting how long a cat would spend in the shelter was a feed forward neural network. It did a bad job at predicting new observations, with a mean absolute error of 57.68 and R^2 score of -0.04.

I think this model can be improved by including a few more variables not present in the data set. For instance, I believe measuring how extroverted a cat is on some kind of scale would help improve model accuracy. Based on my personal experiences, cats who are more outgoing tend to get adopted more than ones who are shy. Also, knowing whether or not a cat was active may have helped in building our models as there may be an imbalance in the number of people who prefer cats that are active and those who prefer cats who are not.

The main point of this project was to get practice building models in Python using the scikit-learn library since most of my machine learning experience has been in R. Also, I wanted to get a better understanding of how neural networks are constructed and applied to data sets. Before working on this project I was under the impression that all kinds of neural networks can be fit to the same data set. However, in my research of neural network implementations, I learned that different neural network architectures solve different problems. The data set and problem I chose to solve made it difficult to find relevant neural network architectures. After several hours of reading articles, journals, online forums, and watching YouTube videos, I decided that a feed forward neural network was the best one to try for my project.

Although I did not achieve a great model performance, I am content with the experience I gained building models in Python, using unsupervised learning, and getting a better understanding of neural networks and their applications all on my own.