

# population

April 15, 2023

```
[1]: import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
import scipy.stats as st
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from datetime import date, datetime

%matplotlib inline
```

```
[2]: df_out = pd.read_pickle('df_out.pkl')
df_out_with_breeds_info = pd.read_pickle('df_out_with_breeds_info.pkl')
df_out.info()
df_out_with_breeds_info.info()
df_out.head()
```

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 149511 entries, 0 to 149510

Data columns (total 41 columns):

#	Column	Non-Null Count	Dtype
0	index	149511 non-null	Int64
1	Animal ID	149511 non-null	string
2	Name	106260 non-null	string
3	Outcome DateTime	149511 non-null	datetime64[ns]
4	Outcome MonthYear	149511 non-null	string
5	Date of Birth	149511 non-null	datetime64[ns]
6	Outcome Type	149485 non-null	string
7	Outcome Subtype	68443 non-null	string
8	Animal Type	149511 non-null	string
9	Sex upon Outcome	149509 non-null	string
10	Age upon Outcome	149465 non-null	string
11	Breed	149511 non-null	string
12	Color	149511 non-null	string

```

13 Intake MonthYear      147980 non-null string
14 Intake DateTime      147980 non-null datetime64[ns]
15 Found Location       147980 non-null string
16 Intake Type          147980 non-null string
17 Intake Condition     147980 non-null string
18 Sex upon Intake      147978 non-null string
19 Age upon Intake      147979 non-null string
20 Years in animal center 147980 non-null Float64
21 Colors (count)       149511 non-null Int64
22 Color 0              149511 non-null string
23 Color 1              79869 non-null string
24 Color 0 R            135638 non-null Float64
25 Color 0 G            135638 non-null Float64
26 Color 0 B            135638 non-null Float64
27 Color 0 H            135638 non-null Float64
28 Color 0 S            135638 non-null Float64
29 Color 0 V            135638 non-null Float64
30 Color 1 R            78596 non-null Float64
31 Color 1 G            78596 non-null Float64
32 Color 1 B            78596 non-null Float64
33 Color 1 H            78596 non-null Float64
34 Color 1 S            78596 non-null Float64
35 Color 1 V            78596 non-null Float64
36 Age upon Outcome (years) 149465 non-null Float64
37 Male                 149509 non-null boolean
38 Female               149509 non-null boolean
39 NeuteredOrSpayed    149509 non-null boolean
40 Adopted              149485 non-null boolean
dtypes: Float64(14), Int64(2), boolean(4), datetime64[ns](3), string(18)
memory usage: 45.6 MB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 149511 entries, 0 to 149510
Data columns (total 57 columns):
#   Column              Non-Null Count  Dtype
---  -
0   index               149511 non-null Int64
1   Animal ID           149511 non-null string
2   Name                106260 non-null string
3   Outcome DateTime    149511 non-null datetime64[ns]
4   Outcome MonthYear   149511 non-null string
5   Date of Birth       149511 non-null datetime64[ns]
6   Outcome Type        149485 non-null string
7   Outcome Subtype     68443 non-null string
8   Animal Type         149511 non-null string
9   Sex upon Outcome    149509 non-null string
10  Age upon Outcome    149465 non-null string
11  Breed               149511 non-null string
12  Color               149511 non-null string

```

13	Intake MonthYear	147980	non-null	string
14	Intake DateTime	147980	non-null	datetime64[ns]
15	Found Location	147980	non-null	string
16	Intake Type	147980	non-null	string
17	Intake Condition	147980	non-null	string
18	Sex upon Intake	147978	non-null	string
19	Age upon Intake	147979	non-null	string
20	Years in animal center	147980	non-null	Float64
21	Colors (count)	149511	non-null	Int64
22	Color 0	149511	non-null	string
23	Color 1	79869	non-null	string
24	Color 0 R	135638	non-null	Float64
25	Color 0 G	135638	non-null	Float64
26	Color 0 B	135638	non-null	Float64
27	Color 0 H	135638	non-null	Float64
28	Color 0 S	135638	non-null	Float64
29	Color 0 V	135638	non-null	Float64
30	Color 1 R	78596	non-null	Float64
31	Color 1 G	78596	non-null	Float64
32	Color 1 B	78596	non-null	Float64
33	Color 1 H	78596	non-null	Float64
34	Color 1 S	78596	non-null	Float64
35	Color 1 V	78596	non-null	Float64
36	Age upon Outcome (years)	149465	non-null	Float64
37	Male	149509	non-null	boolean
38	Female	149509	non-null	boolean
39	NeuteredOrSpayed	149509	non-null	boolean
40	Adopted	149485	non-null	boolean
41	BreedsInfoName	138419	non-null	object
42	Breed (catalog)	138419	non-null	string
43	Breed Group AKC	138419	non-null	string
44	Breed Group CKC	138419	non-null	string
45	Breed Group UKC	138419	non-null	string
46	CKC Subgroup	138414	non-null	string
47	height_low_inches	138419	non-null	Float64
48	height_high_inches	138419	non-null	Float64
49	average height	138419	non-null	Float64
50	weight_low_lbs	138419	non-null	Float64
51	weight_high_lbs	138419	non-null	Int64
52	average weight	138419	non-null	Float64
53	Lifespan Low	138415	non-null	Int64
54	Lifespan High	138415	non-null	Int64
55	average lifespan	138419	non-null	Float64
56	Est. lifespan remaining	138408	non-null	float64

dtypes: Float64(20), Int64(5), boolean(4), datetime64[ns](3), float64(1), object(1), string(23)

memory usage: 66.3+ MB

```
[2]:      index Animal ID      Name      Outcome DateTime Outcome MonthYear \
0    61546   A659834  Dudley  2013-10-01 09:31:00      Oct 2013
1    50833   A664235   <NA>  2013-10-01 10:39:00      Oct 2013
2    93227   A664236   <NA>  2013-10-01 10:44:00      Oct 2013
3   109856   A664237   <NA>  2013-10-01 10:44:00      Oct 2013
4    12697   A664223   Moby  2013-10-01 11:03:00      Oct 2013

      Date of Birth      Outcome Type Outcome Subtype Animal Type Sex upon Outcome \
0    2013-07-23      Adoption      Foster      Dog      Neutered Male
1    2013-09-24      Transfer      Partner      Cat      Unknown
2    2013-09-24      Transfer      Partner      Cat      Unknown
3    2013-09-24      Transfer      Partner      Cat      Unknown
4    2009-09-30  Return to Owner      <NA>      Dog      Neutered Male

      ... Color 1 G Color 1 B Color 1 H Color 1 S Color 1 V \
0    ...      <NA>      <NA>      <NA>      <NA>      <NA>
1    ...      1.0      1.0      0.0      0.0      1.0
2    ...      1.0      1.0      0.0      0.0      1.0
3    ...      1.0      1.0      0.0      0.0      1.0
4    ...      <NA>      <NA>      <NA>      <NA>      <NA>

      Age upon Outcome (years)      Male Female NeuteredOrSpayed Adopted
0              0.166667      True  False      True      True
1              0.019231     False  False      False     False
2              0.019231     False  False      False     False
3              0.019231     False  False      False     False
4              4.0      True  False      True      False
```

[5 rows x 41 columns]

```
[3]: def population(start, end):
      loc_intake = (df_out['Intake DateTime'].isna() | (df_out['Intake DateTime']_
      ↪< end))
      loc_outcome = (df_out['Outcome DateTime'].isna() | (df_out['Outcome_
      ↪DateTime'] >= start))
      return df_out.loc[loc_intake & loc_outcome]
```

```
[4]: start = datetime(2014, 1, 1)
      end = datetime(2023, 4, 1)
      offset = datetime(start.year, start.month, start.day)

      window_start = 'Window start'
      window_end = 'Window end'
      feature_1 = 'Intakes'
      feature_1x_classes = df_out['Intake Type'].unique()
      feature_1y_classes = df_out['Intake Condition'].unique()
```

```

feature_1x = [f'Intakes Types ({feature_class})' for feature_class in
    ↪feature_1x_classes]
feature_1y = [f'Intakes Conditions ({feature_class})' for feature_class in
    ↪feature_1y_classes]
feature_2 = 'Outcomes'
feature_2x_classes = df_out['Outcome Type'].unique()
feature_2x = [f'Outcomes ({feature_class})' for feature_class in
    ↪feature_2x_classes]
feature_3 = 'Breeds (unique)'
feature_4 = 'Animals (count)'
feature_5a = 'Years in animal center (mean)'
feature_5b = 'Years in animal center (std dev)'

df_populations = pd.DataFrame()

df_populations[window_start] = pd.Series(dtype=df_out["Outcome DateTime"].dtype)
df_populations[window_end] = pd.Series(dtype=df_out["Outcome DateTime"].dtype)

df_populations.set_index(window_start)

df_populations[feature_1] = pd.Series(dtype=int)
for feature in feature_1x:
    df_populations[f'{feature} (Absolute)'] = pd.Series(dtype=int)
    df_populations[f'{feature} (Relative)'] = pd.Series(dtype=float)
df_populations = df_populations.copy()
for feature in feature_1y:
    df_populations[feature] = pd.Series(dtype=int)
    df_populations[f'{feature} (Absolute)'] = pd.Series(dtype=int)
    df_populations[f'{feature} (Relative)'] = pd.Series(dtype=float)
df_populations = df_populations.copy()

df_populations[feature_2] = pd.Series(dtype=int)
for feature in feature_2x:
    df_populations[feature] = pd.Series(dtype=int)
    df_populations[f'{feature} (Absolute)'] = pd.Series(dtype=int)
    df_populations[f'{feature} (Relative)'] = pd.Series(dtype=float)
df_populations = df_populations.copy()

df_populations[feature_3] = pd.Series(dtype=int)

df_populations = df_populations.copy()

while offset != end:
    offset_next = datetime(offset.year + (1 if offset.month == 12 else 0),
    ↪(offset.month % 12) + 1, offset.day)

    df_populations.at[offset, window_start] = offset

```

```

df_populations.at[offset, window_end] = offset_next

animals = population(offset, offset_next)

intakes = animals.loc[(animals['Intake DateTime'] >= offset) &
↳(animals['Intake DateTime'] < offset_next)]
df_populations.at[offset, feature_1] = intakes.shape[0]
for feature_class, count in intakes['Intake Type'].value_counts().items():
    df_populations.at[offset, f'Intakes Types ({feature_class})']
↳(Absolute)'] = count
    df_populations.at[offset, f'Intakes Types ({feature_class})']
↳(Relative)'] = count / intakes.shape[0]
    for feature_class, count in intakes['Intake Condition'].value_counts().
↳items():
        df_populations.at[offset, f'Intakes Conditions ({feature_class})']
↳(Absolute)'] = count
        df_populations.at[offset, f'Intakes Conditions ({feature_class})']
↳(Relative)'] = count / intakes.shape[0]

outcomes = animals.loc[(animals['Outcome DateTime'] >= offset) &
↳(animals['Outcome DateTime'] < offset_next)]
df_populations.at[offset, feature_2] = outcomes.shape[0]
for feature_class, count in outcomes['Outcome Type'].value_counts().items():
    df_populations.at[offset, f'Outcome Types ({feature_class})']
↳(Absolute)'] = count
    df_populations.at[offset, f'Outcome Types ({feature_class})']
↳(Relative)'] = count / outcomes.shape[0]

df_populations.at[offset, feature_3] = len(animals['Breed'].unique())
df_populations.at[offset, feature_4] = animals.shape[0]

df_populations.at[offset, feature_5a] = animals["Years in animal center"].
↳mean()
df_populations.at[offset, feature_5b] = animals["Years in animal center"].
↳std()

offset = offset_next

```

```

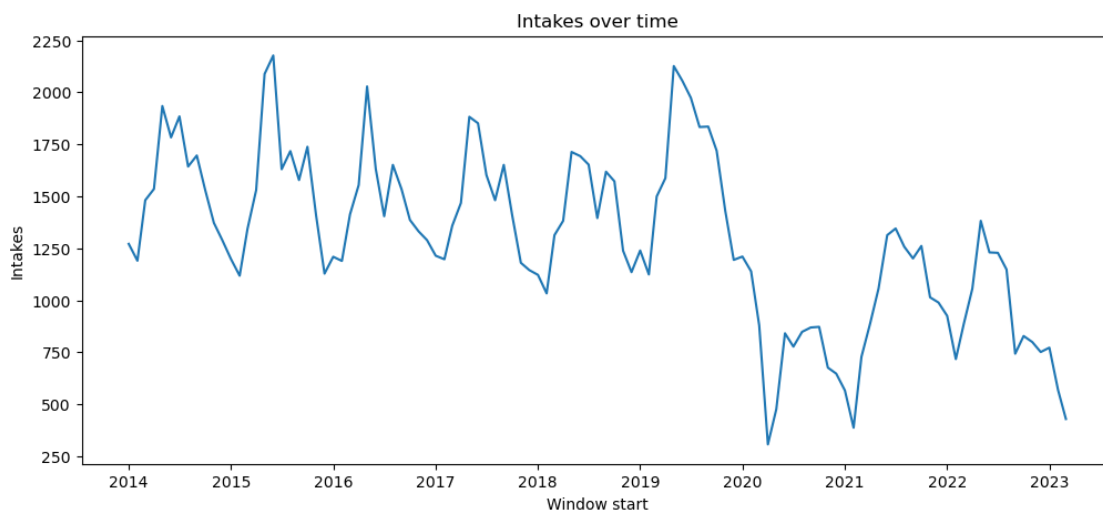
[5]: df_populations['Month'] = pd.Series(dtype=int)
df_populations['Year'] = pd.Series(dtype=int)
for index in df_populations.index:
    df_populations.at[index, 'Month'] = df_populations.at[index, window_start].
↳month
    df_populations.at[index, 'Year'] = df_populations.at[index, window_start].
↳year

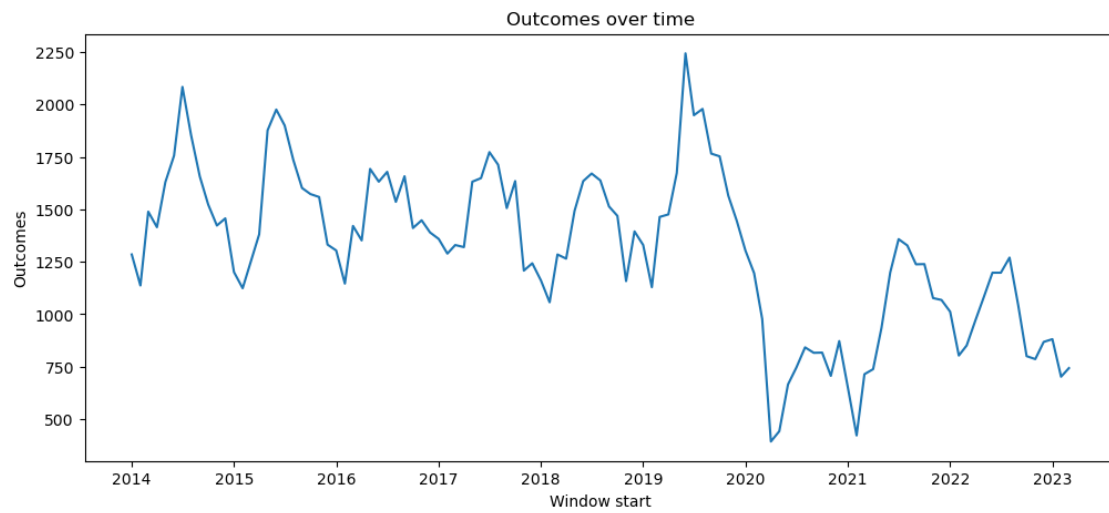
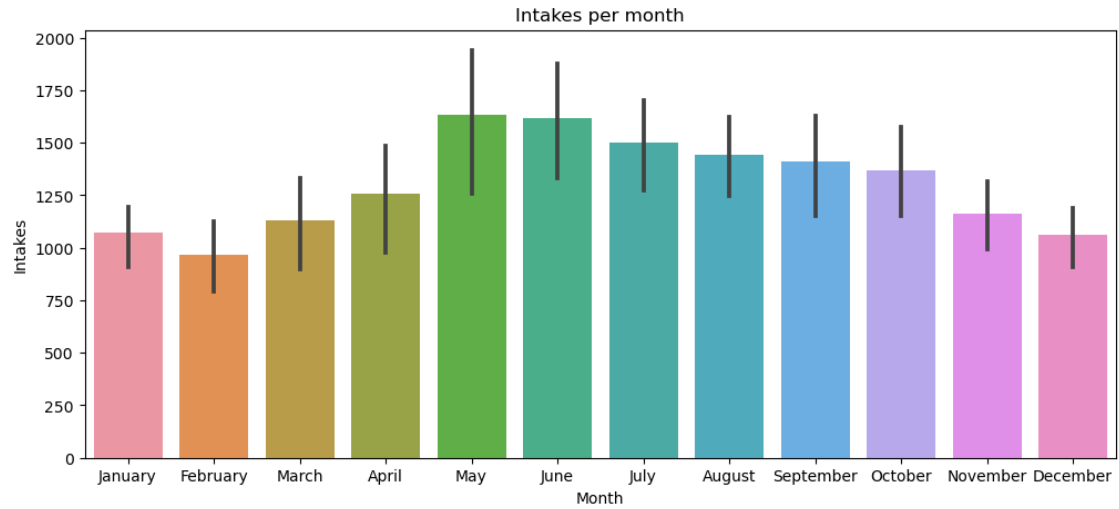
```

```
[6]: def populationCharts(feature):
    plt.figure(figsize=(12, 5))
    sns.lineplot(
        data=df_populations,
        x=window_start,
        y=feature
    )
    plt.title(f'{feature} over time')
    plt.show()

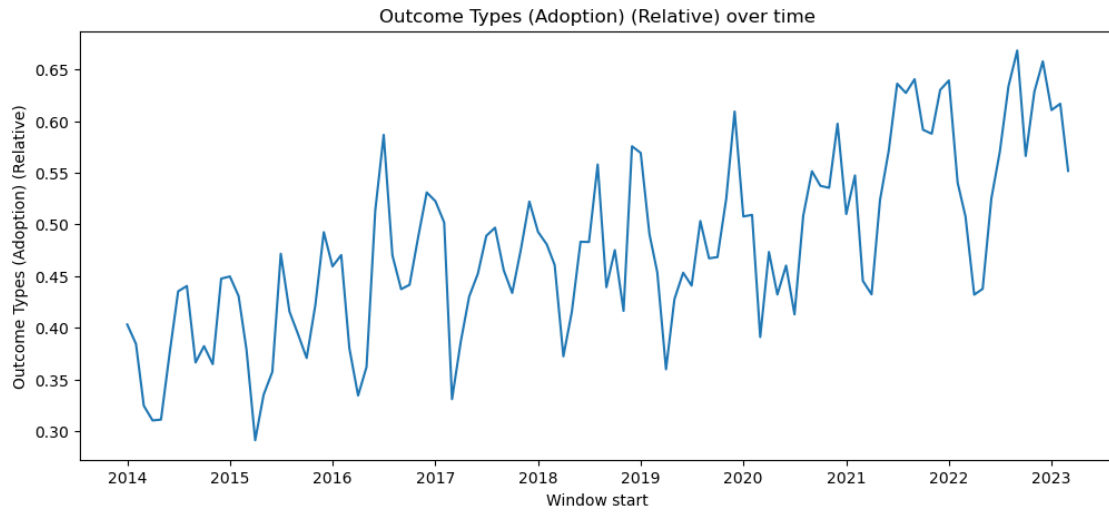
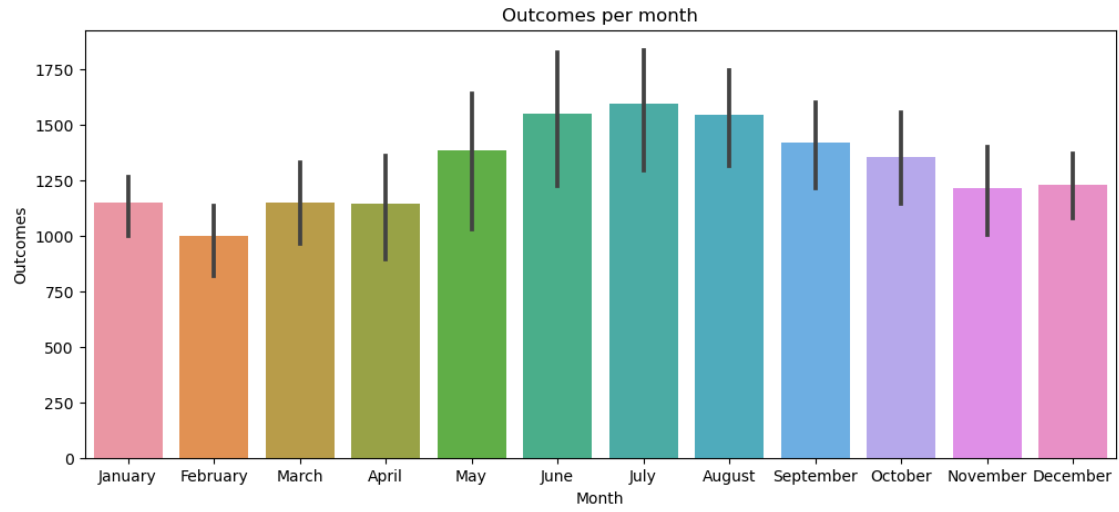
    plt.figure(figsize=(12, 5))
    sns.barplot(
        data=df_populations,
        x='Month',
        y=feature,
    )
    plt.xticks(range(12), ["January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"])
    plt.title(f'{feature} per month')
    plt.show()

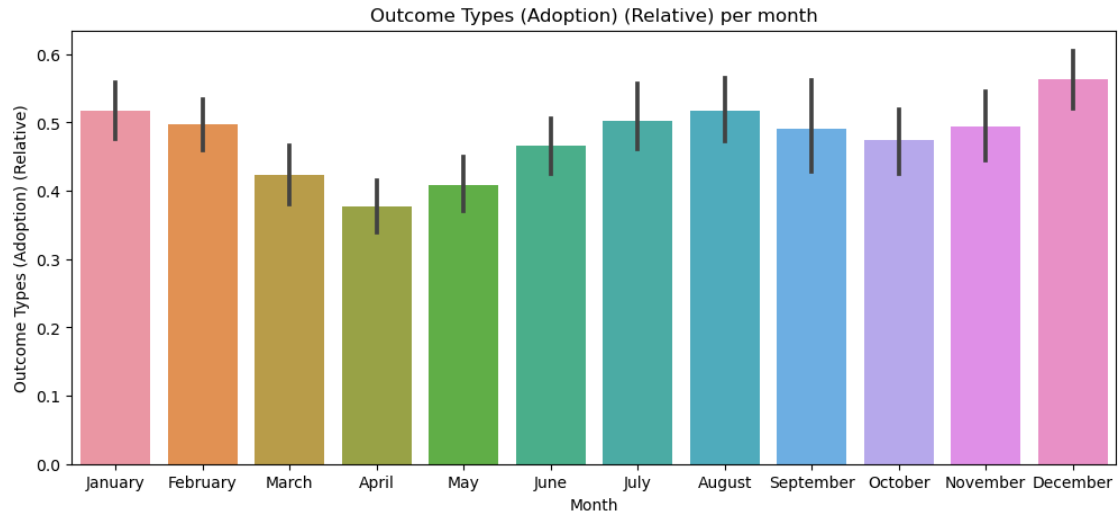
populationCharts('Intakes')
populationCharts('Outcomes')
populationCharts('Outcome Types (Adoption) (Relative)')
```











```
[7]: df_populations.Year.head(20)
```

```
[7]: 2014-01-01    2014.0
      2014-02-01    2014.0
      2014-03-01    2014.0
      2014-04-01    2014.0
      2014-05-01    2014.0
      2014-06-01    2014.0
      2014-07-01    2014.0
      2014-08-01    2014.0
      2014-09-01    2014.0
      2014-10-01    2014.0
      2014-11-01    2014.0
      2014-12-01    2014.0
      2015-01-01    2015.0
      2015-02-01    2015.0
      2015-03-01    2015.0
      2015-04-01    2015.0
      2015-05-01    2015.0
      2015-06-01    2015.0
      2015-07-01    2015.0
      2015-08-01    2015.0
      Name: Year, dtype: float64
```

```
[8]: def adoptionCorr(feature, hue):
      sns.scatterplot(
          data=df_populations,
          x=feature,
          y='Outcome Types (Adoption) (Relative)',
```

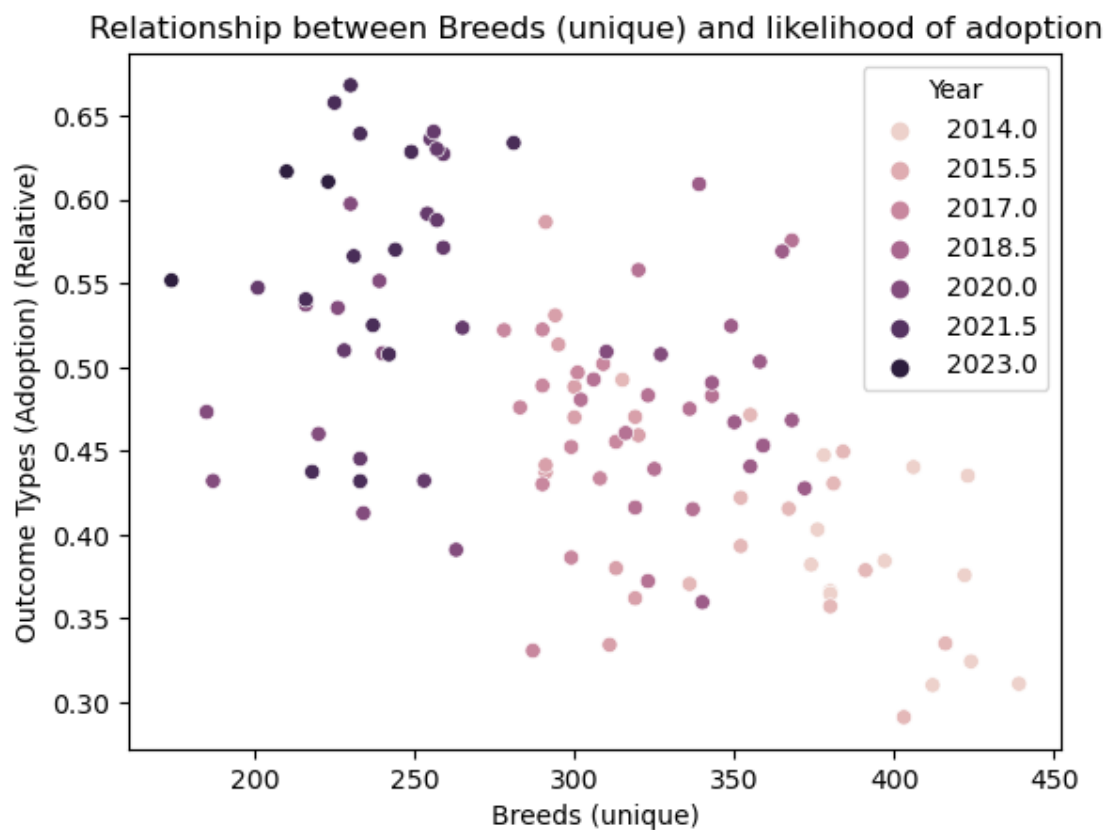
```

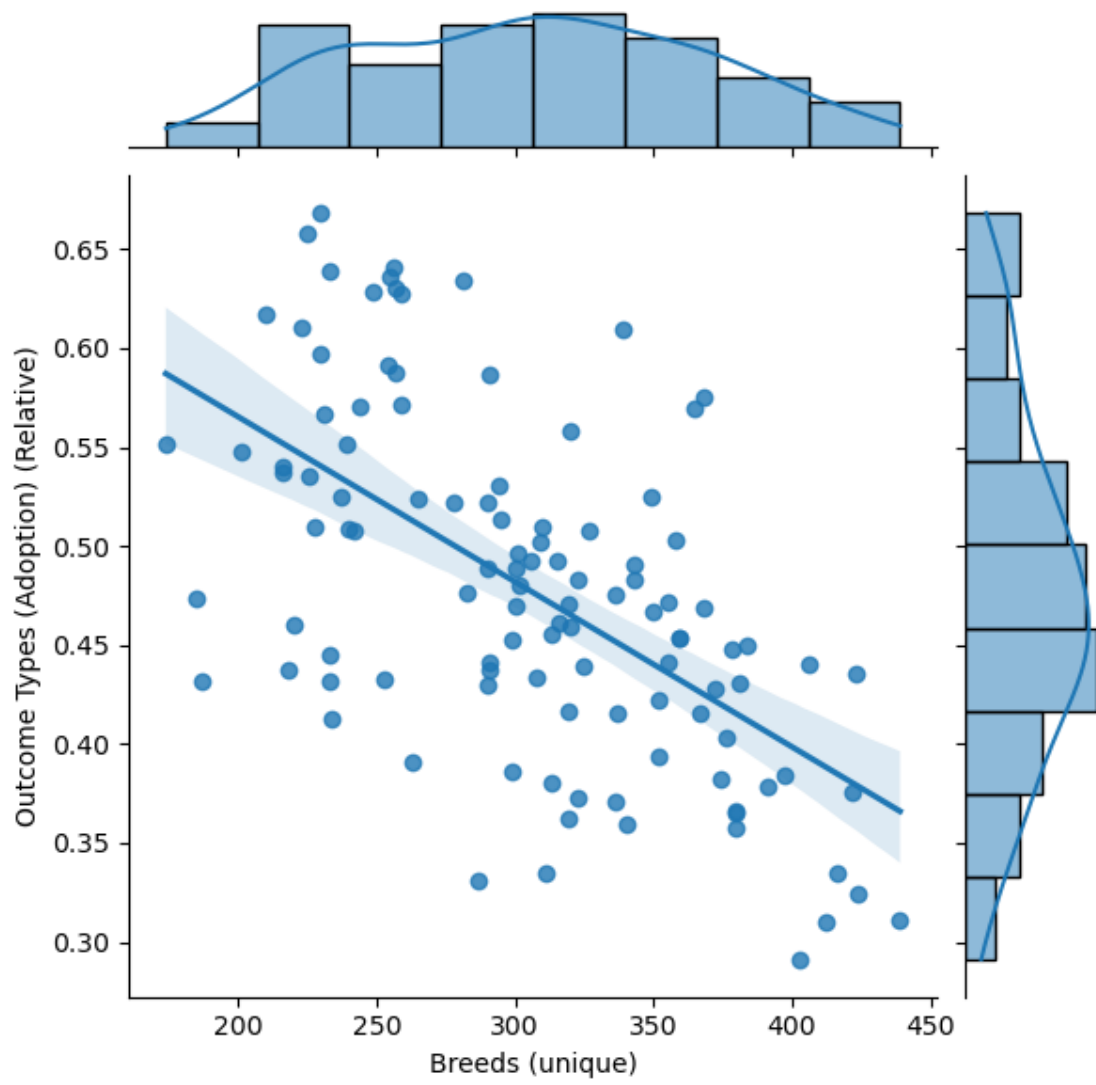
        hue=hue
    )
    plt.title(f"Relationship between {feature} and likelihood of adoption")
    plt.show()

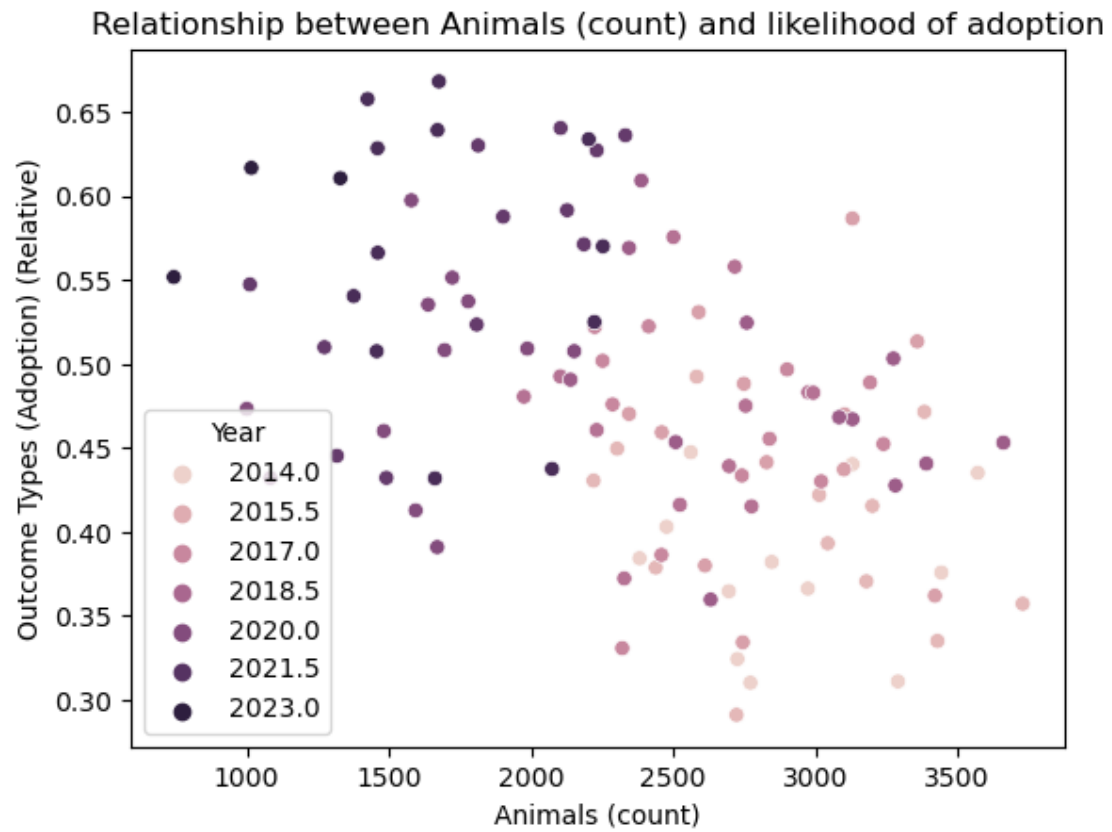
    try:
        sns.jointplot(
            data=df_populations,
            x=feature,
            y='Outcome Types (Adoption) (Relative)',
            kind='reg'
        )
        plt.show()
    except: pass

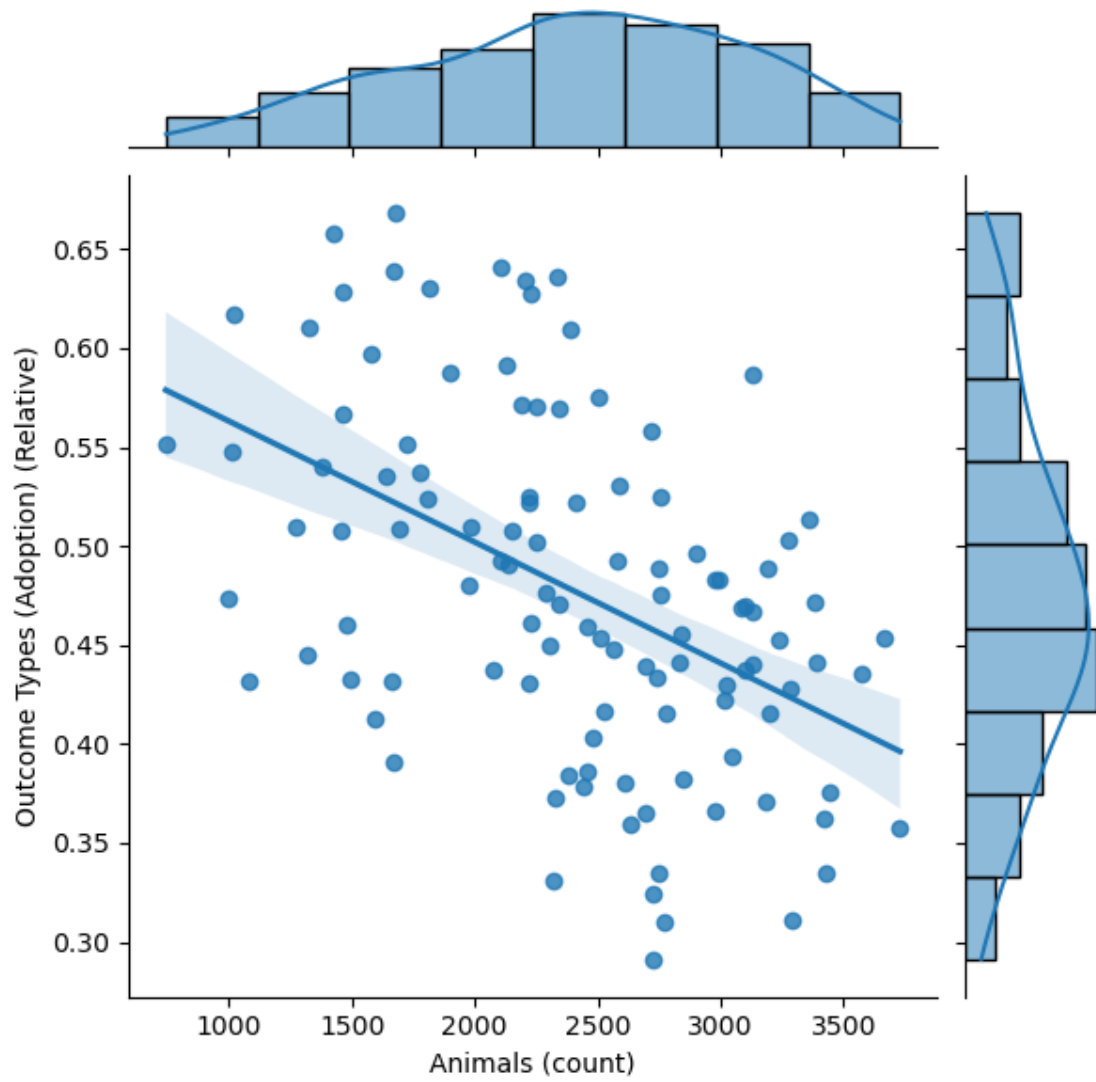
adoptionCorr(feature_3, 'Year')
adoptionCorr(feature_4, 'Year')
adoptionCorr(feature_5a, 'Year')
adoptionCorr(window_start, None)

```









Relationship between Years in animal center (mean) and likelihood of adoption

