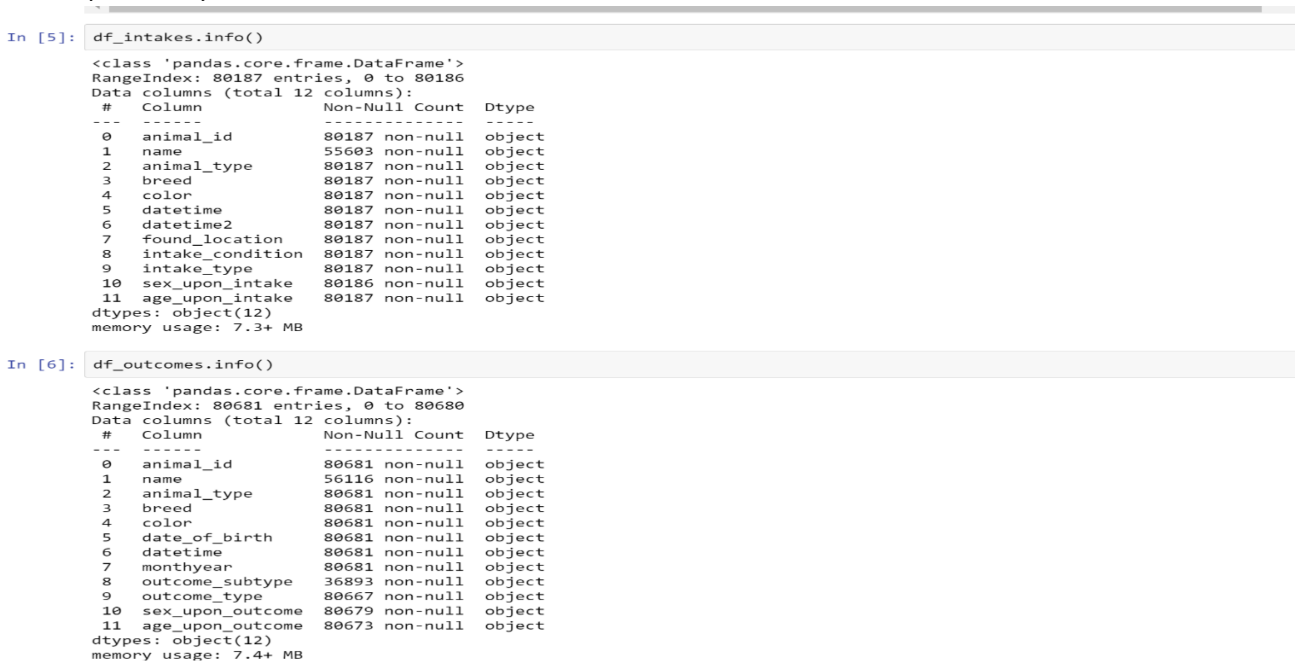
Project Report (Data Sc. Tools - 1)

Austin's Animal Shelter Analysis

--Submitted By: Anshul Dabas

1. **Project's Jupyter Notebook Code Github Link**:
   https://git.cs.du.edu/ansdabas/machine-learning-projects-msds/-/tree/master/Data-Sc-Tools-1

2. **Dataset Motivation**:
   I wanted to work with an animal shelter data to apply data science methods that can help the shelter management to solve any existing problem or improvise their methods for maximum successful adoptions rate in order to give better life to the needful animals.
   In the USA, Austin Animal Centre is the largest no-kill animals' shelter. I found their animals intake and animals outcome datasets on Kaggle on following link:
   https://www.kaggle.com/aaronschlegel/austin-animal-center-shelter-intakes-and-outcomes
   These datasets helped me in my purpose of applying data science techniques to impact adoption rates in an animals' shelter related data. Also, the datasets were publicly available as there was not any license related norm. I downloaded 2 csv files from above mentioned link of Kaggle to get animals_intake data and animals_outcomes data with all relevant features.
   Metadata -
   The following screenshot provides all the details of the metadata for both datasets with self-explanatory features' name.



```
In [5]: df_intakes.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 80187 entries, 0 to 80186
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   animal_id         80187 non-null  object
 1   name              55603 non-null  object
 2   animal_type       80187 non-null  object
 3   breed             80187 non-null  object
 4   color             80187 non-null  object
 5   datetime          80187 non-null  object
 6   datetime2         80187 non-null  object
 7   found_location    80187 non-null  object
 8   intake_condition  80187 non-null  object
 9   intake_type       80187 non-null  object
 10  sex_upon_intake   80186 non-null  object
 11  age_upon_intake   80187 non-null  object
dtypes: object(12)
memory usage: 7.3+ MB
```

```
In [6]: df_outcomes.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 80681 entries, 0 to 80680
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   animal_id         80681 non-null  object
 1   name              56116 non-null  object
 2   animal_type       80681 non-null  object
 3   breed             80681 non-null  object
 4   color             80681 non-null  object
 5   date_of_birth     80681 non-null  object
 6   datetime          80681 non-null  object
 7   monthyear         80681 non-null  object
 8   outcome_subtype   36893 non-null  object
 9   outcome_type      80667 non-null  object
 10  sex_upon_outcome  80679 non-null  object
 11  age_upon_outcome  80673 non-null  object
dtypes: object(12)
memory usage: 7.4+ MB
```

3. **Research Question:**

   Can we predict, if a dog/cat can be adopted from USA's largest shelter home in 1 month, given their color, breed, sex, intake age, intake condition and intake type.

   This question will solve the problem related to early successful adoption rate at an animal shelter. It will help to understand the contributing features to early animals' adoptions. Further, it will benefit the center officials to manage their animals population effectively. For eg – They can manage arrangements for early adopted kind of animals differently than other animals to facilitate efficient resources for both kind of animals.

   Input - features of animal
   Output – binary indicator(0/1) if that animal can be adopted in a month

   My analysis will basically lead to binary classification modelling and its interpretation.

4. **Literature Review:**

   There has been great work done by others using these datasets. Couple of projects were – to explore pit bull breed vs other breeds in dogs to break stereotypes/myths about them that lead pit bull to be a less desirable dog, another one – to analyze how people react to abandoned animals straying around(mostly dogs and cats).

   My work is basically targeted for animal center officials/managers to effectively manage the resources for animals(dog/cats) according to their adoption rates.

   They can use my model to test/predict which dogs/cats will get an early adoption and which ones are less likely to get adopted. Accordingly, they can organize and allocate their resources efficiently for more and less likely adoption animals.

5. **Workflow:**

   a. Data Collection:

      I collected the data from both animal_intake and animal_outcome csv files into pandas dataframes.

   b. Data Cleaning:

      Both the dataframe (df_intakes and df_outcomes) had missing values and incorrect datatypes etc.

      I began with data cleaning of df_outcomes dataframe:

      First, Missing values - In df_outcomes, 5 features (name, outcome_subtype, outcome_type, sex_upon_outcome and age_upon_outcome) had null/missing values. For my analysis, I required only outcome_type feature for tracking the

adoption related outcomes. Also, column outcome_subtypes had ~50% null values. Logically, dropping this column made sense as it had many null values and would not be used further in the analysis.
Further, the name column was imputed by replacing null values with 'no name' string value. This is because I was not sure if name column could be required further in the analysis and dropping 24565 rows can impact the dataset adversely.
Next, the rows with null values for outcome_type would not be useful for the analysis as it was my output column/variable. Also, there were only 14 rows with outcome_type as null value. So, I dropped rows with null value for outcome_type.
Then, there were only 2 rows with missing values for sex_upon_outcome. As they were few rows with missing values, I dropped them as well. Lastly, age_upon_outcome column had 6 missing values. I imputed these values with their breed's age mean value after datatype conversion explained below for this feature. However, one row still had null value for age because no mean value was possible due to no other animal in that same breed(Guinea). So, I dropped that single row with null value for age.
Finally, the df_outcomes dataframe was free from any null/missing values.

Secondly, datatype conversions for required columns was addressed:
The age values were of string datatype, I changed them to int values as number of days as age value of animals. Basically, I created a customized function(age_in_days_int) to convert age string value into age int values as number days. This helped to generate age values as numbers with unit as days for all the animals. Next, I changed the datatype of datetime and monthyear columns from string object to datatime objects. Also, I realized they were duplicated columns which led to dropping of monthyear column from the dataframe df_outcomes.

Finally, I dropped all the rows that had animal_type other than dog/cat as per my analysis requirement.

Next, data cleaning of df_intakes dataframe:

First, Missing values – There were missing values in only two features(name, sex_upon_intake). Similar to df_outcomes name missing values case, I again imputed the null values in name column with 'no name' string value due to the same reason as mentioned in df_outcomes case.
Then, I dropped the only single row with sex_upon_intake as null value.
Therefore, I had df_intakes with no existing missing values.

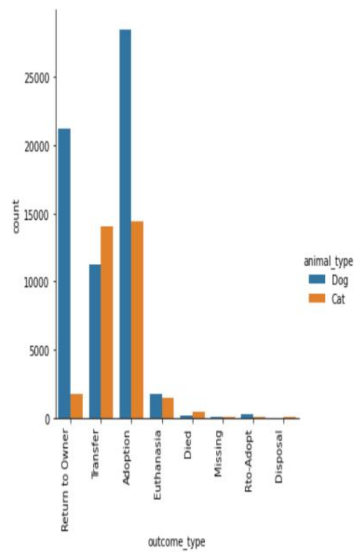
Second, datatype conversions:
The datetime and datetime2 columns were duplicate columns. I dropped datatime2 column to avoid duplicates in the dataframe. Then, I converted the datetime column datatype to datetime object and age_upon_intake to int as number of days. I used same method to convert age_upon_intake values as in the case of df_outcomes by using age_in_days_int function.

Finally, I dropped all the rows that had animal_type other than dog/cat as per my analysis requirement.

c. Data Condensation:
To prepare a final dataset as input for the model, I merged both the datasets by using inner join on animal id variable. There were many repeated columns in the merged dataframe df_merge, so I dropped all the duplicate columns and renamed the columns with appropriate names.
Then, I removed all the 44 duplicated rows from the df_merge dataframe as it had impacted the further analysis.

d. Exploratory Data Analysis:
Further, I used some data visualization techniques to explore my merged dataframe to understand the data. The below visualizations gave me ideas about any updates required in the dataset, understand the data distribution and analyze the data.

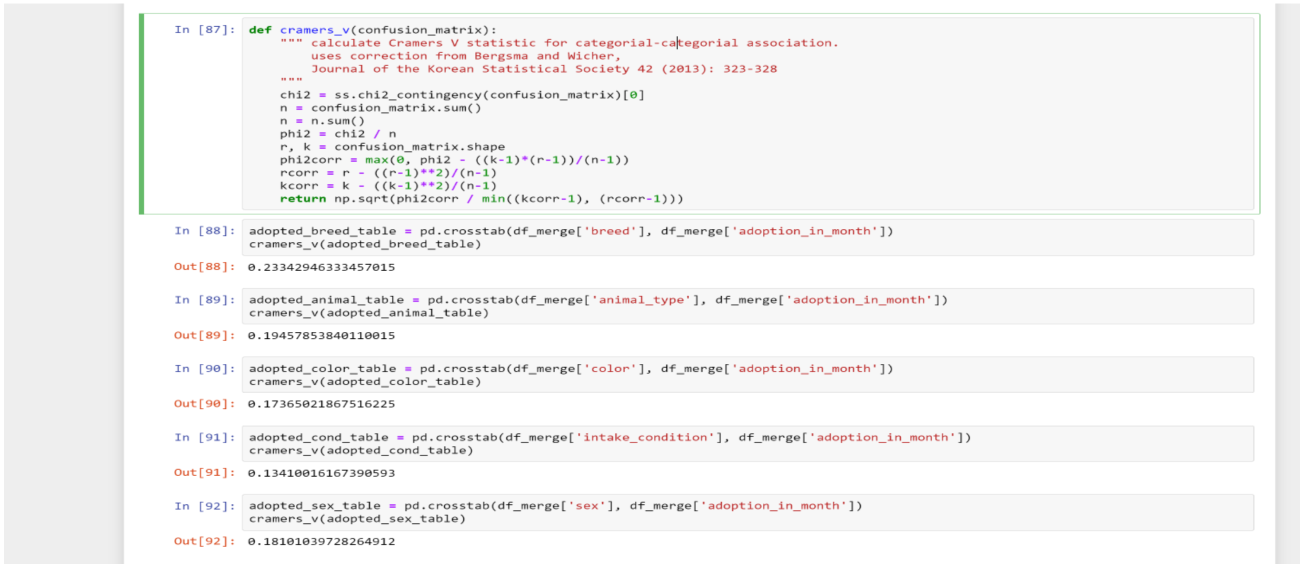
```
Out[53]: <seaborn.axisgrid.FacetGrid at 0x2789595e7b8>
```



This plot helped to understand that we need to focus on outcome_type as ‘Return To Owner’, ‘Adoption’, Rto-Adopt’ to study the adoption success rates. So, as a

result all the other outcome_type was removed from the dataframe. Also, it was interpreted from the above plot that adopted or returned to owner animals had more dogs in comparison with cats by significant difference. However, cats in the transfer category were more than dogs.
Further, when I tried to use category plot on breed and color, I realized there were more than 100 categories which would further make my modelling/analysis inconsistent. So, I discovered top 15 breeds for both dogs and cats, then I updated my dataframe with only top 30 breeds (dogs and cat). Similarly, I did the same for color categories by updating df_merge with only top 30 colors for both dogs and cats (15 categories each).
Then, I checked if any other nominal feature had numerous categories and found that only found_location column had the same issue. Also, logically, the adoption is not impacted by found location of the animal. So, I dropped this column as well. However, some key points to note from found_location were that majority of the animals were found in Austin, Texas, that makes sense as the shelter is located in Austin, Texas. Generally, it's expected to have majority of animals from the shelter's city. Also, found location for an animal does not contributes to its adoption or other outcome values.So, it was dropped from the final dataset.
Finally, I had all the nominal features with relevant number of categories.
Then, I created a column 'adoption_in_month', that will be my output variable further. First, I calculated number of days an animal had been in the shelter by using datetime_intake and datetime_outcome. Then, by using a threshold of 30 days to create a binary column – 'adoption_in_month'.
Further, I dropped all the columns that were irrelevant or had no contributions to the output variable.

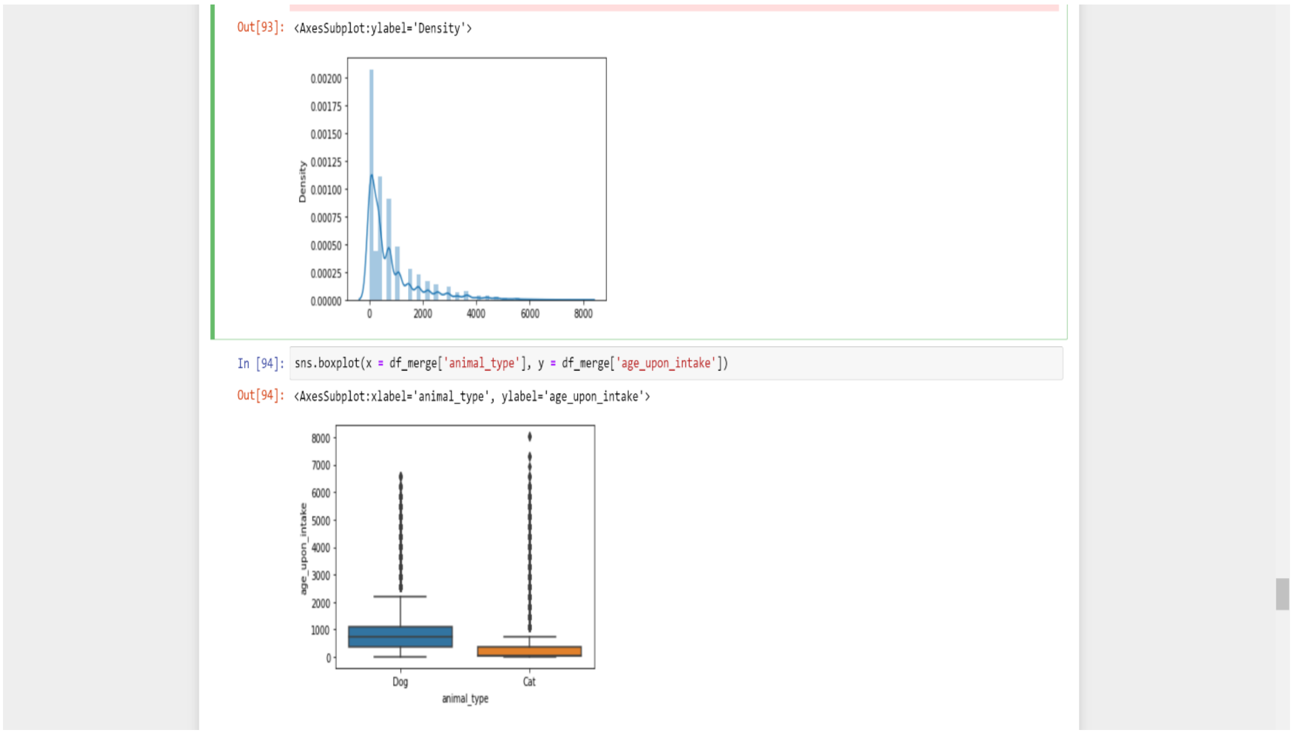
<u>Calculating correlations:</u>



```
In [87]: def cramers_v(confusion_matrix):
             """ calculate Cramers V statistic for categorial-categorial association.
                 uses correction from Bergsma and Wicher,
                 Journal of the Korean Statistical Society 42 (2013): 323-328
             """
             chi2 = ss.chi2_contingency(confusion_matrix)[0]
             n = confusion_matrix.sum()
             n = n.sum()
             phi2 = chi2 / n
             r, k = confusion_matrix.shape
             phi2corr = max(0, phi2 - ((k-1)*(r-1))/(n-1))
             rcorr = r - ((r-1)**2)/(n-1)
             kcorr = k - ((k-1)**2)/(n-1)
             return np.sqrt(phi2corr / min((kcorr-1), (rcorr-1)))

In [88]: adopted_breed_table = pd.crosstab(df_merge['breed'], df_merge['adoption_in_month'])
         cramers_v(adopted_breed_table)

Out[88]: 0.23342946333457015

In [89]: adopted_animal_table = pd.crosstab(df_merge['animal_type'], df_merge['adoption_in_month'])
         cramers_v(adopted_animal_table)

Out[89]: 0.19457853840110015

In [90]: adopted_color_table = pd.crosstab(df_merge['color'], df_merge['adoption_in_month'])
         cramers_v(adopted_color_table)

Out[90]: 0.17365021867516225

In [91]: adopted_cond_table = pd.crosstab(df_merge['intake_condition'], df_merge['adoption_in_month'])
         cramers_v(adopted_cond_table)

Out[91]: 0.13410016167390593

In [92]: adopted_sex_table = pd.crosstab(df_merge['sex'], df_merge['adoption_in_month'])
         cramers_v(adopted_sex_table)

Out[92]: 0.18101039728264912
```

Surprisingly, there was poor correlation between any predictor feature and the outcome variable. So, any linear relationship does not make sense. As, all the features instead of age were nominal, I used cramers_v to calculate required correlations.

Further, I used box plot and dist plot to understand the data distribution of age_upon_intake feature, as it was the only numerical variable in the dataset.

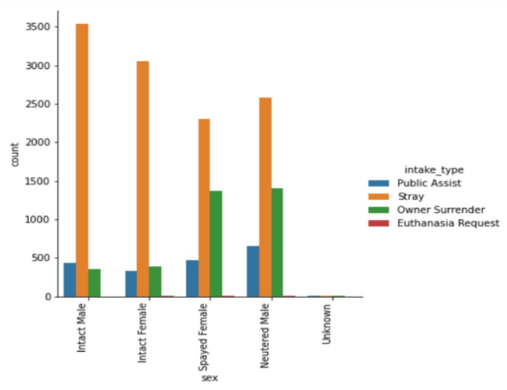


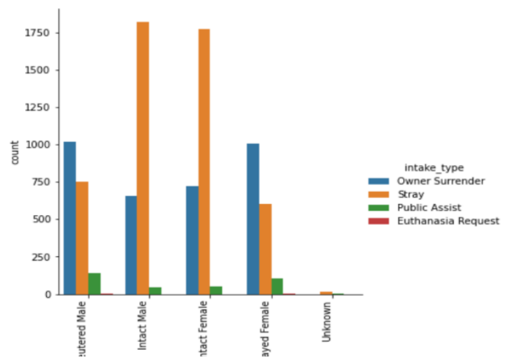
```
Out[93]: <AxesSubplot:ylabel='Density'>
```

```
In [94]: sns.boxplot(x = df_merge['animal_type'], y = df_merge['age_upon_intake'])
Out[94]: <AxesSubplot:xlabel='animal_type', ylabel='age_upon_intake'>
```

From the above age plots, it was interpreted that age_upon_intake data distribution was skewed as per normality. Median was around 800 days for dogs and all the animals' age fall under 1000 days. There were outlier data points as well after 2000 days. Also, cats' intake age (median around 400days) was less than dogs' age. Even, outliers were more in the cats than dogs.

Next, I plotted intake_type vs sex of the animals to check the category distribution of both the feature variables.



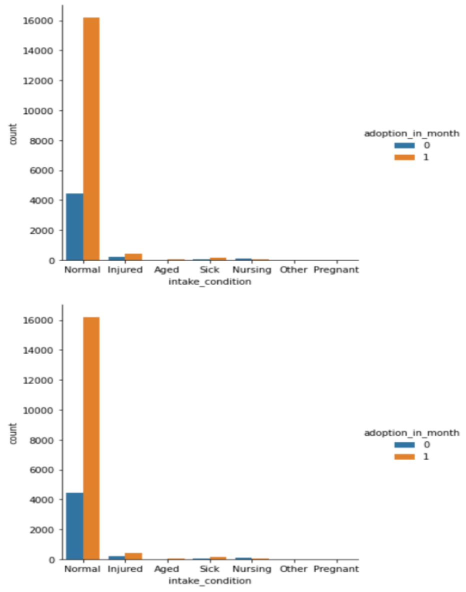


In the case of dogs, the intake type was stray across all sex categories. Also, owners surrendered dogs significantly that belonged to category of spayed female and neutered males.

In the case of cats, most of the cats were surrendered by owners in case of spayed female and neutered males. Also, cats were significantly stray upon intake.
Overall, most of the dogs and cats were stray or surrendered by the owner across all sex categories.

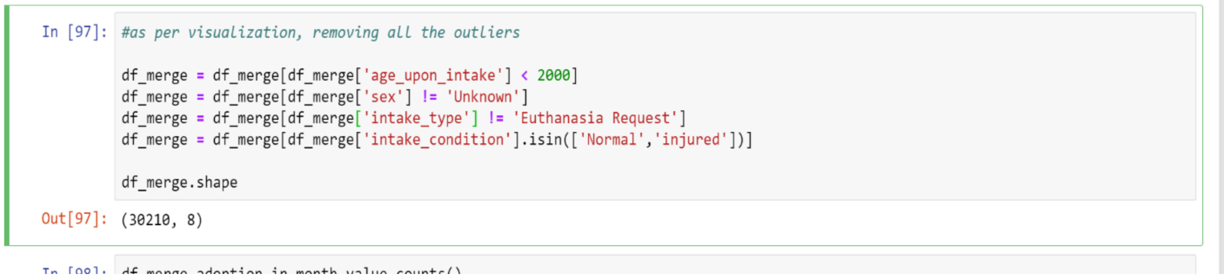
Next, I plotted to understand the category distribution of the intake condition of the animals across single month adoption outcome variable.

```
Out[96]: <seaborn.axisgrid.FacetGrid at 0x278a84d3a90>
```



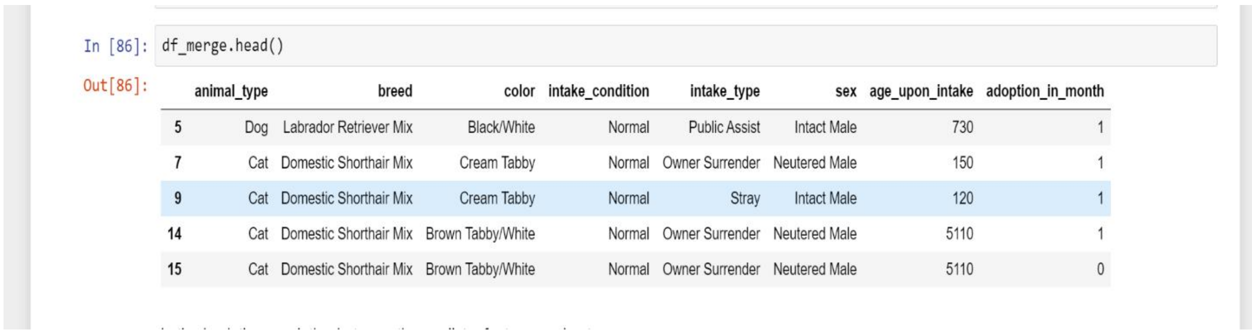
According to the above plot, by majority, normal condition animals have greater chances to get adopted in a month significantly. All the other categories stand as outlier except injured condition.

As a result of all the distribution plot above, outlier categories/values were identified and removed as follows:



```
In [97]: #as per visualization, removing all the outliers

         df_merge = df_merge[df_merge['age_upon_intake'] < 2000]
         df_merge = df_merge[df_merge['sex'] != 'Unknown']
         df_merge = df_merge[df_merge['intake_type'] != 'Euthanasia Request']
         df_merge = df_merge[df_merge['intake_condition'].isin(['Normal','injured'])]

         df_merge.shape

Out[97]: (30210, 8)

In [98]: df_merge.adoption_in_month.value_counts()
```

Final cleansed dataset that was used for further data modelling and analysis without any outliers:

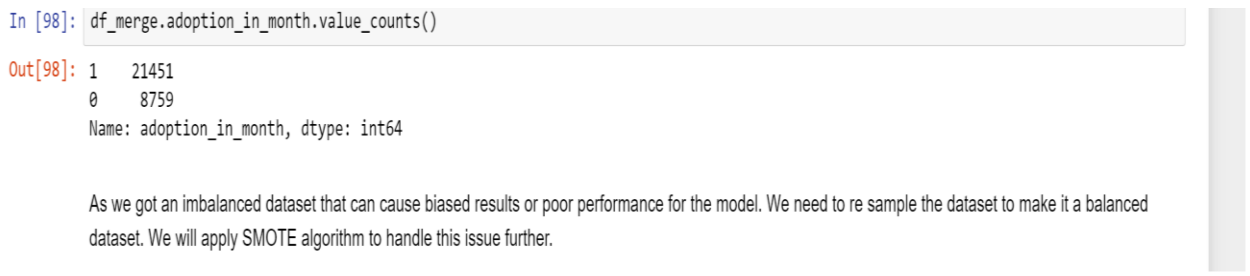


```
In [86]: df_merge.head()

Out[86]:
```

	animal_type	breed	color	intake_condition	intake_type	sex	age_upon_intake	adoption_in_month
5	Dog	Labrador Retriever Mix	Black/White	Normal	Public Assist	Intact Male	730	1
7	Cat	Domestic Shorthair Mix	Cream Tabby	Normal	Owner Surrender	Neutered Male	150	1
9	Cat	Domestic Shorthair Mix	Cream Tabby	Normal	Stray	Intact Male	120	1
14	Cat	Domestic Shorthair Mix	Brown Tabby/White	Normal	Owner Surrender	Neutered Male	5110	1
15	Cat	Domestic Shorthair Mix	Brown Tabby/White	Normal	Owner Surrender	Neutered Male	5110	0

e. Data Modelling:

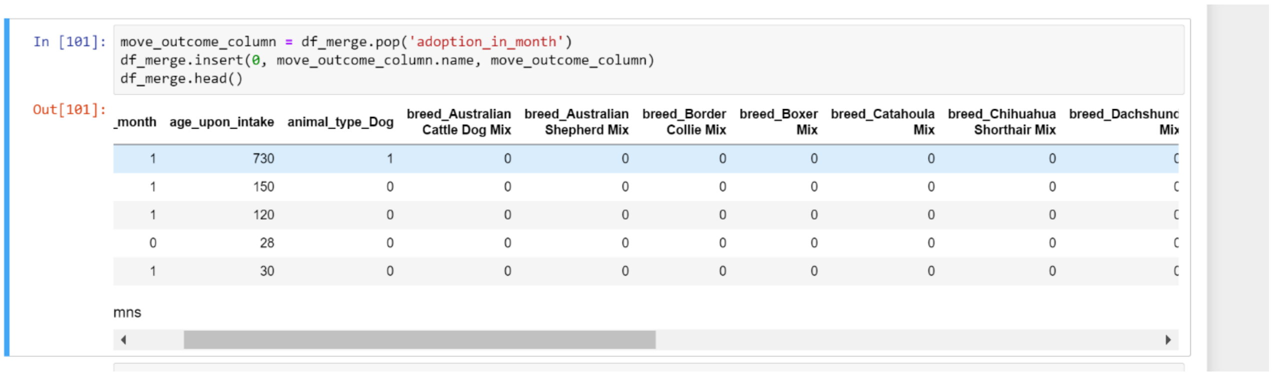
At first, my dataset was an imbalanced dataset for applying any classification model. So, it was important to resample my data using some techniques to have a balanced dataset before training the model.



```
In [98]: df_merge.adoption_in_month.value_counts()

Out[98]: 1    21451
         0     8759
         Name: adoption_in_month, dtype: int64
```

As we got an imbalanced dataset that can cause biased results or poor performance for the model. We need to re sample the dataset to make it a balanced dataset. We will apply SMOTE algorithm to handle this issue further.

As all the features were nominal in the dataset except age_upon_intake, I applied one-hot encoding for further data modeling.
As a result, there were total 72 columns and 30210 rows in the final encoded input dataset.



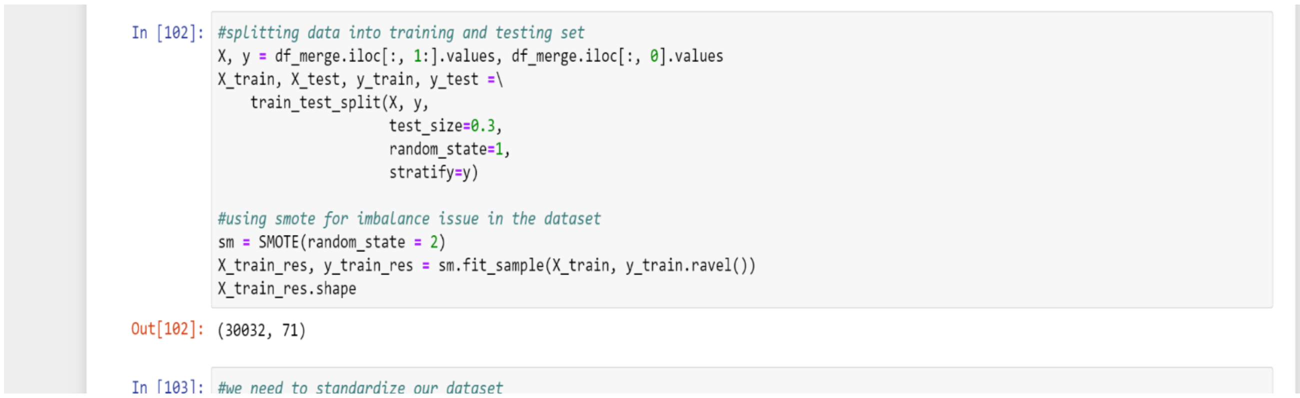
```
In [101]: move_outcome_column = df_merge.pop('adoption_in_month')
          df_merge.insert(0, move_outcome_column.name, move_outcome_column)
          df_merge.head()
```

Out[101]:

_month	age_upon_intake	animal_type_Dog	breed_Australian Cattle Dog Mix	breed_Australian Shepherd Mix	breed_Border Collie Mix	breed_Boxer Mix	breed_Catahoula Mix	breed_Chihuahua Shorthair Mix	breed_Dachshunc Mi
1	730	1	0	0	0	0	0	0	C
1	150	0	0	0	0	0	0	0	C
1	120	0	0	0	0	0	0	0	C
0	28	0	0	0	0	0	0	0	C
1	30	0	0	0	0	0	0	0	C

mns

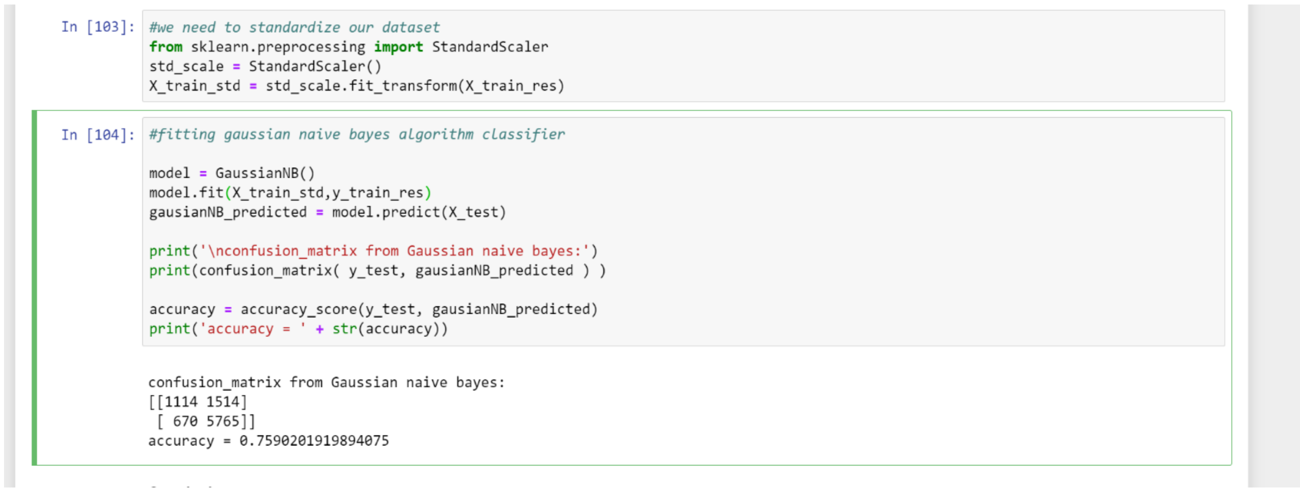
Then, the dataset was split into training and testing data (70%:30% ratio) for both input and output variables. Further, smote algorithm was applied on the training data to oversample the minority class to result in balanced dataset.



```
In [102]: #splitting data into training and testing set
          X, y = df_merge.iloc[:, 1:].values, df_merge.iloc[:, 0].values
          X_train, X_test, y_train, y_test =\
              train_test_split(X, y,
                               test_size=0.3,
                               random_state=1,
                               stratify=y)

          #using smote for imbalance issue in the dataset
          sm = SMOTE(random_state = 2)
          X_train_res, y_train_res = sm.fit_sample(X_train, y_train.ravel())
          X_train_res.shape

Out[102]: (30032, 71)

In [103]: #we need to standardize our dataset
```

Finally, binary classifier – **Gaussian Naïve bayes algorithm** was applied to the normalized scaled training data. We applied this model because it works great for nominal normalized training data for classification. For a Naive Bayes classifier, categorical values are the easiest to deal with as it uses probability for doing its predictive analysis.

Also, the data was normally scaled that helped the gaussian naive bayes algorithm to perform well.



```
In [103]: #we need to standardize our dataset
          from sklearn.preprocessing import StandardScaler
          std_scale = StandardScaler()
          X_train_std = std_scale.fit_transform(X_train_res)

In [104]: #fitting gaussian naive bayes algorithm classifier

          model = GaussianNB()
          model.fit(X_train_std,y_train_res)
          gausianNB_predicted = model.predict(X_test)

          print('\nconfusion_matrix from Gaussian naive bayes:')
          print(confusion_matrix( y_test, gausianNB_predicted ) )

          accuracy = accuracy_score(y_test, gausianNB_predicted)
          print('accuracy = ' + str(accuracy))


          confusion_matrix from Gaussian naive bayes:
          [[1114 1514]
           [ 670 5765]]
          accuracy = 0.7590201919894075
```

Naive bayes classifier gave ~76% accuracy. Also, we can further improve the accuracy of by using feature selection methods, hyperparameter tuning and different sampling methods.

f. Conclusion:

The gaussian naive bayes classification algorithm performed fairly well with the given predictor features of animals. This answered our reseach question that we can predict a month adoption of an animal with given factors significantly. However, there is a scope of performance improvement to get better prediction significance. Also, other classification algorithm can be tested to get better performance. Therefore, we can predict if a dog/cat can be adopted in a month of their intake in the shelter given their breed, color, intake_type, intake_condition, sex and age upon intake. These predictions can further be used by the shelter management for efficient resource management for the animals. For example – if shelter is on low budget to order food for all the animals for next 2-3 months in advance, they can manage on ordering food efficiently with low finance, given which ones are highly likely to get adopted from the shelter. Similarly, theer can be various other cases to apply this model and analysis as a usecase for shelter and animals welfare.