Are Pitbulls one of the most problematic dog Breeds?

> by: Anna Bellizzi & Nicole George

Introduction

We have all heard that "pitbulls are one of the most dangerous dog breeds", but we want to see the data behind it and prove that every dog is different, just like every human is different. We want to figure out what is the likelihood of a bite incident occurring in a dog's lifetime based on size, breed and age. We want to compare this data to people's perception of how dangerous specific breeds are regardless of size, age, or breed. The machine learning aspect will be using intelligence to predict the likelihood of a dog biting and hopefully help more people be able to adopt dogs that are better suited for them, capable of handling, and ultimately have every dog end up in a Forever Home.

Below is the imports that will be needed for running the dataframes, visualizations, and machine learning predictions.

In [9]:
```
!pip install pyreadstat
```

```
Requirement already satisfied: pyreadstat in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (1.2.0)
Requirement already satisfied: pandas>=1.2.0 in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (from pyreadstat) (1.4.4
)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (from pandas>=1
.2.0->pyreadstat) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (from pandas>=1.2.0->pyre
adstat) (2022.1)
Requirement already satisfied: numpy>=1.18.5 in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (from pandas>=1.2.0->pyr
eadstat) (1.21.5)
Requirement already satisfied: six>=1.5 in /Users/potatofamily/opt/anaconda3/lib/python3.9/site-packages (from python-dateutil>=2.8.1-
>pandas>=1.2.0->pyreadstat) (1.12.0)
```

In [10]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import pyreadstat
from matplotlib import rcParams
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
import tarfile
from sklearn.linear_model import LinearRegression

# allow output to span multiple output lines in the console
pd.set_option('display.max_columns', 500)

# switch to seaborn default stylistic parameters
# see the useful https://seaborn.pydata.org/tutorial/aesthetics.html
sns.set()
sns.set_context('paper') # 'talk' for slightly larger

# change default plot size
rcParams['figure.figsize'] = 9,7
```

In [11]:
```
# required packages according to PetFinder
#import pandas as pd
#import numpy as np
#import seaborn as sns
#import matplotlib.pyplot as plt
#import sklearn.metrics as metrics
#from sklearn.model_selection import train_test_split
#from sklearn.model_selection import cross_val_score
#from sklearn.neighbors import KNeighborsClassifier
#from sklearn.metrics import accuracy_score
#from sklearn.metrics import confusion_matrix
#from sklearn.metrics import cohen_kappa_score
#from sklearn.linear_model import LogisticRegression
#from sklearn.naive_bayes import GaussianNB
#from sklearn.ensemble import RandomForestClassifier
#from sklearn.compose import ColumnTransformer
#from sklearn.pipeline import Pipeline
#from sklearn.impute import SimpleImputer
#from sklearn.preprocessing import StandardScaler, OneHotEncoder
#from sklearn.model_selection import GridSearchCV
```

In [12]:
```
#from google.colab import drive
#drive.mount('/content/drive')
# /content/drive/Shareddrives/Data Science Project/DogBiteData/ny-dog-bites-2015-2021-CLEAN.csv
#dog bite data

#/content/drive/Shareddrives/Data Science Project/DogBiteData/sf-raw-data-dog-bites-2014-2018-CLEAN.csv
#sf raw dog bite data
```

San Francisco Bite Data 2014-2018

San Francisco Bite Data, 2014-2018 - Total rows: 3863
Exported PDF to Exel via Adobe Acrobat
Removed rows that contained only total by breed sections - 267 rows
Removed header rows that appeard within the doc as a PDF - 475 rows
Split bite severity into two seperate columns - BITE_CODE and BITE_SEVERITY
Retaining both columns for now, both are not necessary since they contain the same infomation
Replaced alpha "U" code in BITE_SEVERITY for UNKNOWN to 9 to match numeric convention of the column
Removed spaces in BITE_SEVERITY alpha codes for ease of search
Removed BITE column because each row value for that column was 1. Each row represents a single event already, column is not necessary.
Total Rows: 3626
Total Cols: 5
Change column names to all lowercase
Imported as CSV - sf-raw-data-dog-bites-2014-2018-CLEAN.csv

NOTE need to correct breed_group, both of these exist in the column: 'GREAT PYRENEES' 'GREAT PYRENEESE'

In [14]:
```
df_sf = pd.read_csv("sf-raw-data-dog-bites-2014-2018-CLEAN.csv")
df_sf.head()
###breed_group may be an unnecessary column:
#breed_group = df_sf.breed_group
#print(breed_group.unique())
#primary_breed = df_sf.primary_breed
#print(primary_breed.unique().size)
print(df_sf.head())
df_sf.info()

df_sf['breed_group'].value_counts().plot(kind='bar')
```

```
     breed_group  primary_breed  bite_code bite_severity gender
0  AFFENPINSCHER  AFFENPINSCHER          0       SNGLNIP      S
1  AFFENPINSCHER  AFFENPINSCHER          1         SNGLB      U
2   AFGHAN HOUND   AFGHAN HOUND          1         SNGLB      N
3  AIREDALE TERR  AIREDALE TERR          0       SNGLNIP      N
4  AIREDALE TERR  AIREDALE TERR          1         SNGLB      N
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3625 entries, 0 to 3624
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   breed_group    3625 non-null   object
 1   primary_breed  3625 non-null   object
 2   bite_code      3625 non-null   int64 
 3   bite_severity  3625 non-null   object
 4   gender         3625 non-null   object
dtypes: int64(1), object(4)
memory usage: 141.7+ KB
<AxesSubplot:>
```

Out[14]:

New York Bite Data 2015-2021

Removed columns Borough, ZipCode, Species
Combined Gender and Spay/Neuter columns to match format of DF_SF gender column
Removed Key column and Date of bite column
Ages modified to standard values, removed alphas and turned numbers into decimal value in years.
Added Bad_Data column to tag potentially bad rows that could be thrown out of the dataset without tossing them out quite yet.
Added breed_group column to match df_sf dataset and clarify primary_breed values that specify a group.
Added Multi-Dog column for entries invloving more than one dog in description. Low numbers, may consider dropping these rows.

Standardized breed column and breed_group column.
About 450 rows out of 22,000 were unable to be categorized, but contain dog descriptions so data was kept and categorized as "unknown".

The data does not include whether one dog is a multi-offender, which may or may not be significant for our analysis.

This dataset can potentially be joined to the SF dataset on the breed_group primary_breed and gender columns.

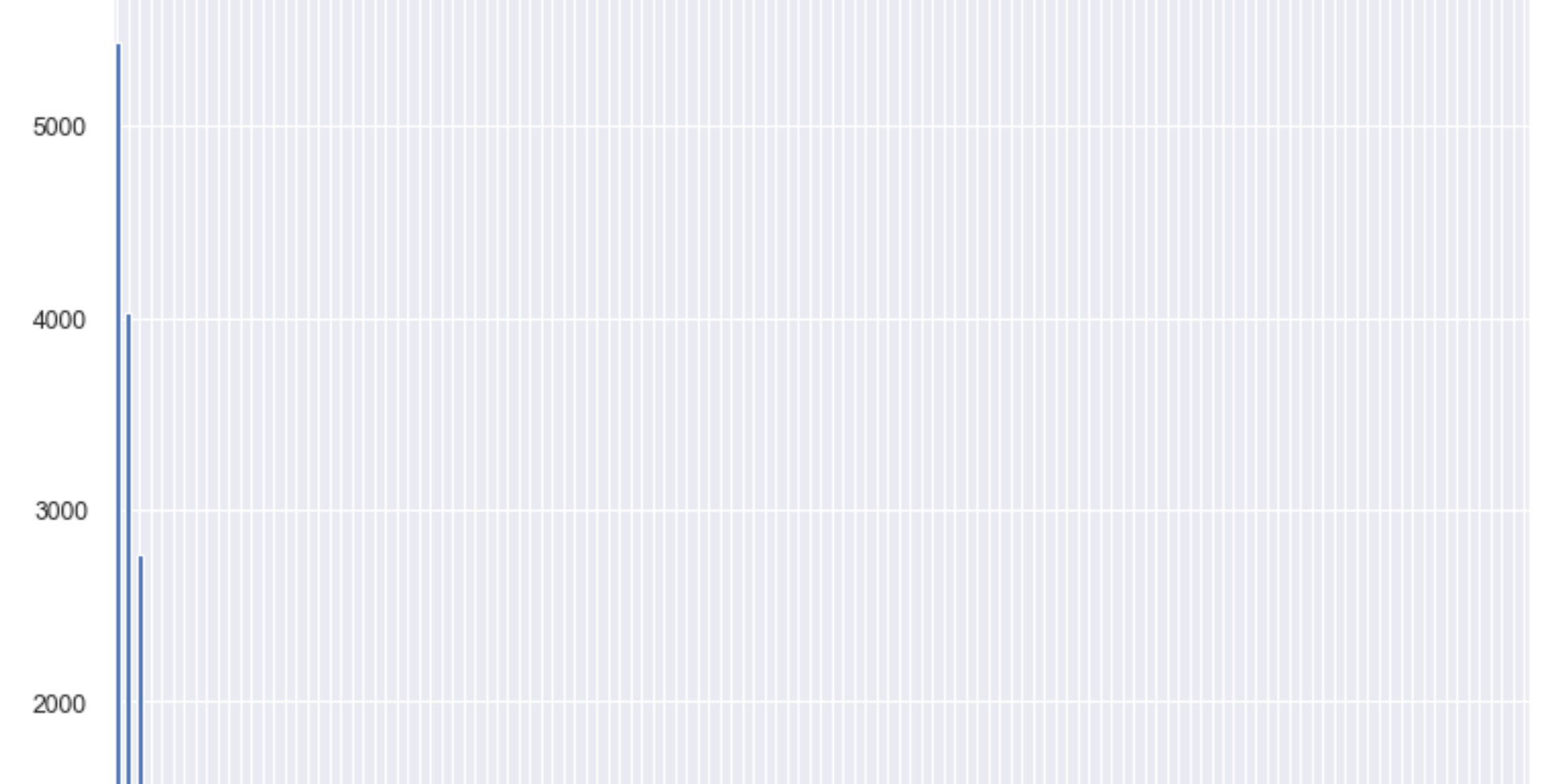
may consider adding a bite_code column with code 1 for all entries. Since an entry here implies an incident it would match up to the existing column in the sf data.

In [15]:
```
df_ny = pd.read_csv("ny-dog-bites-2015-2021-CLEAN.csv")
df_ny.head()
df_ny.info()

df_ny['breed_group'].value_counts().plot(kind='bar')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22663 entries, 0 to 22662
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   breed_group    20445 non-null  object
 1   primary_breed  20445 non-null  object
 2   age            22663 non-null  object
 3   modified_age   22658 non-null  object
 4   gender         22663 non-null  object
 5   multi_dog      22663 non-null  object
 6   bad_data       22663 non-null  object
dtypes: object(7)
memory usage: 1.2+ MB
<AxesSubplot:>
```

Out[15]:



Dog Breeds Enriched

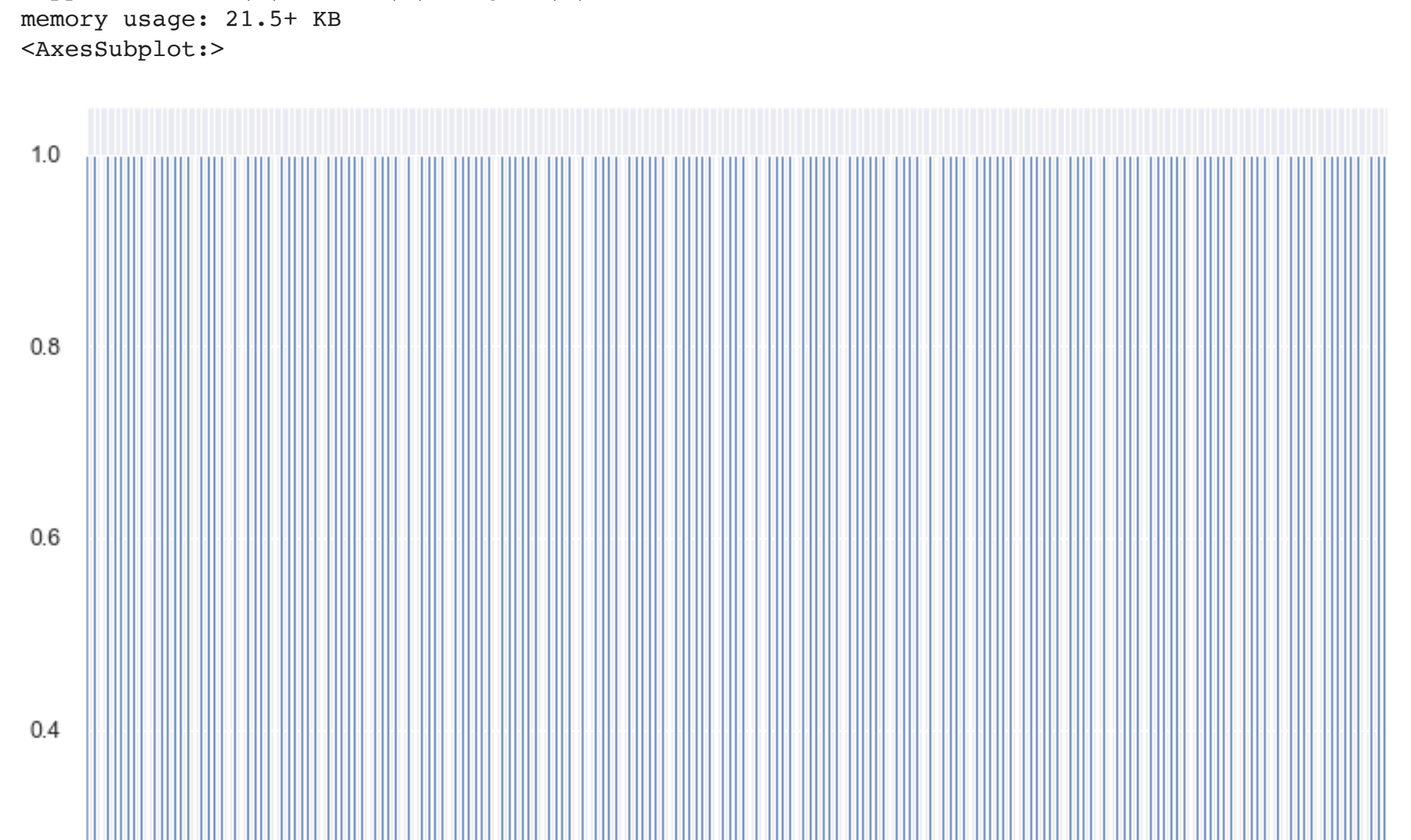
In [16]:
```
df_be = pd.read_csv("dog breeds_enriched_20210503.csv")

df_be.head()
df_be.info()

df_be['Breed'].value_counts().plot(kind='bar')
#Note change name to breed_group to be able to combine all 3 data sets
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Breed               195 non-null    object 
 1   Breed Group AKC     195 non-null    object 
 2   Breed Group CKC     195 non-null    object 
 3   Breed Group UKC     195 non-null    object 
 4   CKC Subgroup        194 non-null    object 
 5   height_low_inches   195 non-null    float64
 6   height_high_inches  195 non-null    float64
 7   average height      195 non-null    float64
 8   weight_low_lbs      195 non-null    float64
 9   weight_high_lbs     195 non-null    int64  
 10  average weight      195 non-null    float64
 11  Lifespan Low        194 non-null    float64
 12  Lifespan High       194 non-null    float64
 13  average lifespan    195 non-null    float64
dtypes: float64(8), int64(1), object(5)
memory usage: 21.5+ KB
<AxesSubplot:>
```

Out[16]:



Compare breed names used in NY and SF data files to those in the enriched breed name dataset

In [18]:
```
sfBreeds = pd.Series(df_sf['breed_group'])
nyBreeds = pd.Series(df_ny['breed_group'])
breeds = pd.Series(df_be['Breed'])

ser = pd.Series(dtype='object')
ser = ser.append(sfBreeds.str.lower())
ser = ser.append(nyBreeds.str.lower())
ser = ser.append(breeds.str.lower())
#print(ser.unique())
```

```
/var/folders/dn/d06ybfk147qdky36xjx27xwh0000gn/T/ipykernel_10515/2460531506.py:6: FutureWarning: The series.append method is deprecate
d and will be removed from pandas in a future version. Use pandas.concat instead.
  ser = ser.append(sfBreeds.str.lower())
/var/folders/dn/d06ybfk147qdky36xjx27xwh0000gn/T/ipykernel_10515/2460531506.py:7: FutureWarning: The series.append method is deprecate
d and will be removed from pandas in a future version. Use pandas.concat instead.
  ser = ser.append(nyBreeds.str.lower())
/var/folders/dn/d06ybfk147qdky36xjx27xwh0000gn/T/ipykernel_10515/2460531506.py:8: FutureWarning: The series.append method is deprecate
d and will be removed from pandas in a future version. Use pandas.concat instead.
  ser = ser.append(breeds.str.lower())
```

In [ ]:

Stanford Dogs Dataset Images of Dog Breeds

In [19]:
```
#Since this is a tarfile I am having trouble opening it, it was only going to be used for aestetic purposes.
#I will continue working on this before the project is due.
#df_st = pd.read_csv('/lists.tar', compression='gzip', header=0, sep=' ', quotechar='"', error_bad_lines=False)
```

Read the data

Initial exploration

[/]how much data is there?

[]how many NA values are in the data?

does the dataset contain much obviously bad data?

what are the types of the columns?

functions like info() and describe() are helpful in this stage

Initial preprocessing and cleaning

remove columns with lots of missing data

remove columns that are useless

remove columns that are not relevant to what you want to do

remove other missing data

Exploration and visualization

histograms of single numeric variables

bar plots of value counts of single categorical variables

grid of scatter plots (numeric variables)

violin/bar plots for categorical/numeric variable pairs

three-variable plots, such as scatterplots with color or shape of points as a third variable, or grouped bar plots plots of data over time (if applicable)

Final preprocessing and cleaning

convert categorical to numeric data

scale data if needed

Machine learning

accuracy

confusion matrix

precision/recall

ROC curve, precision/recall curve (if predictions are probabilities)

MSE, RMSE

R-squared statistic (usually computed on training data)

predicted/actual scatterplot

grid search to tune hyperparameters

feature selection (such as forward feature selection)

learning curve

create training and test sets

train model and make predictions

assess results (classification case)

assess result (regression case)

tuning

cross-validation can be used in both assessment and tuning

assess bias/variance

Merging Data Sets

In [20]:
```
#data= pd.merge(df_sf, df_ny, df_be)
```

Incidents by top 5 breeds

Consider analyzing top 3 breeds from NY data and better categorizing. Many "MIX" and unknown can be identified by breeds listed in the dog breeds_enriched dataset.

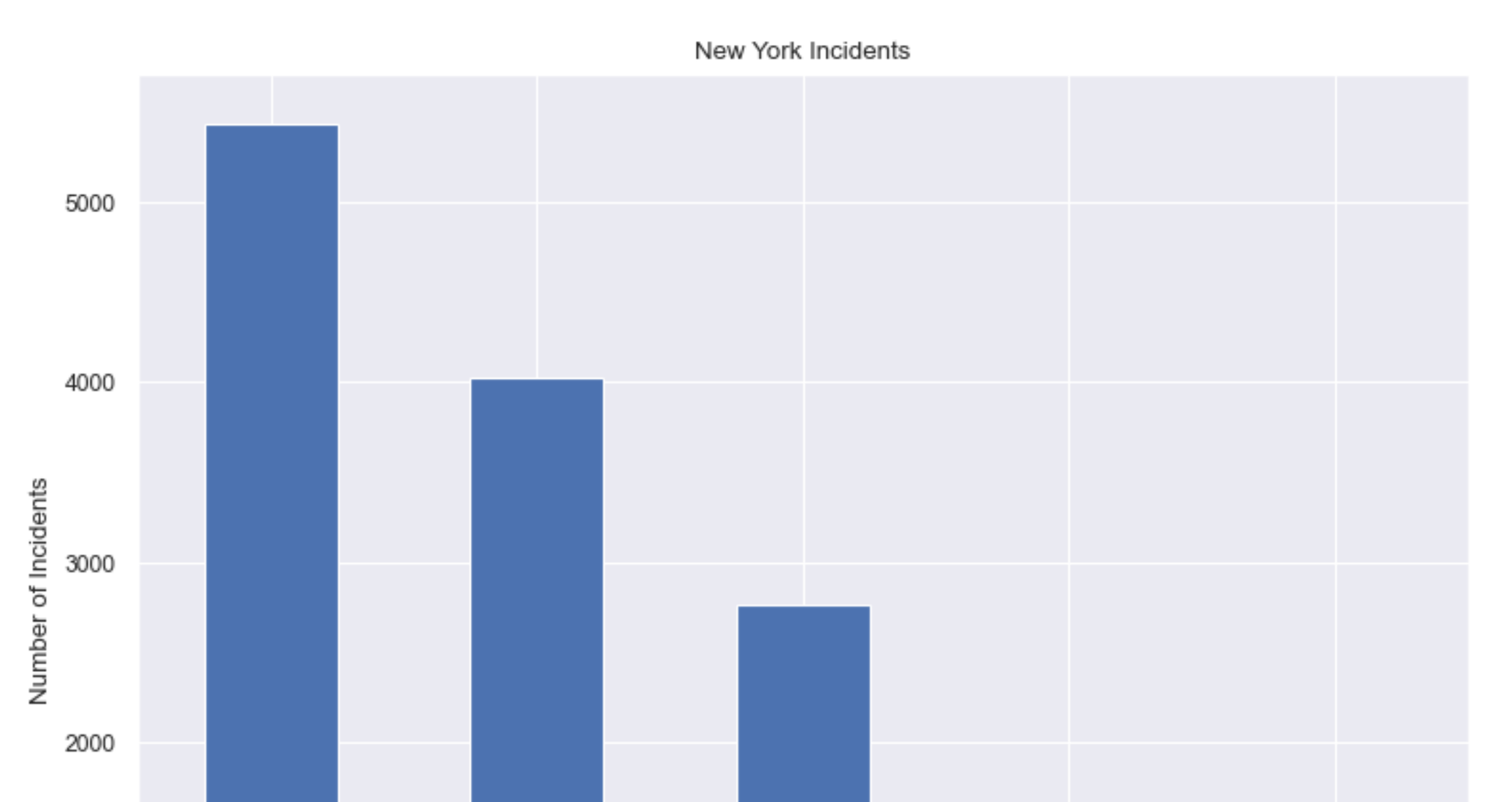
In [21]:
```
sf_inc = df_sf['breed_group'].value_counts().nlargest(5)
ny_inc = df_ny['breed_group'].value_counts().nlargest(5)
print(sf_inc)
print(ny_inc)


sf_inc.plot(kind='bar')
plt.xlabel("Breed Group")
plt.ylabel("Number of Incidents")
plt.title("San Francisco Incidents")

ny_inc.plot(kind='bar')
plt.xlabel("Breed Group")
plt.ylabel("Number of Incidents")
plt.title("New York Incidents")
```

```
CHIHUAHUA        1281
PIT BULL          719
LABRADOR RETR     198
GERM SHEPHERD     198
BOXER              56
Name: breed_group, dtype: int64
PIT MIX       5437
MIX           4029
UNKNOWN       2762
SHIH TZU       732
CHIHUAHUA      648
Name: breed_group, dtype: int64
Text(0.5, 1.0, 'New York Incidents')
```

Out[21]:



As you can see here, the largest breed groups by incidents are chihuahua according to san fransisco and pitmix according to new york.

Machine Learning

We are going to start by using linear regression for the machine learning.

In [22]:
```
X=df_sf[['breed_group']].values
y=df_ny['breed_group'].values

regr=LinearRegression()
regr.fit(X,y)
fit= regr.predict(X)

sns.regplot(x=fit, y='breed_group', data=df_sf)
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/var/folders/dn/d06ybfk147qdky36xjx27xwh0000gn/T/ipykernel_10515/3998173002.py in <module>
      3 
      4 regr=LinearRegression()
----> 5 regr.fit(X,y)
      6 fit= regr.predict(X)
      7 

~/opt/anaconda3/lib/python3.9/site-packages/sklearn/linear_model/_base.py in fit(self, X, y, sample_weight)
    660         accept_sparse = False if self.positive else ["csr", "csc", "coo"]
    661 
--> 662         X, y = self._validate_data(
    663             X, y, accept_sparse=accept_sparse, y_numeric=True, multi_output=True
    664         )

~/opt/anaconda3/lib/python3.9/site-packages/sklearn/base.py in _validate_data(self, X, y, reset, validate_separately, **check_params)
    579                 y = check_array(y, **check_y_params)
    580             else:
--> 581                 X, y = check_X_y(X, y, **check_params)
    582             out = X, y
    583 

~/opt/anaconda3/lib/python3.9/site-packages/sklearn/utils/validation.py in check_X_y(X, y, accept_sparse, accept_large_sparse, dtype,
order, copy, force_all_finite, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_features, y_numeric, estimator)
    962         raise ValueError("y cannot be None")
    963 
--> 964     X = check_array(
    965         X,
    966         accept_sparse=accept_sparse,

~/opt/anaconda3/lib/python3.9/site-packages/sklearn/utils/validation.py in check_array(array, accept_sparse, accept_large_sparse, dtyp
e, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    744                     array = array.astype(dtype, casting="unsafe", copy=False)
    745                 else:
--> 746                     array = np.asarray(array, order=order, dtype=dtype)
    747             except ComplexWarning as complex_warning:
    748                 raise ValueError(

ValueError: could not convert string to float: 'AFFENPINSCHER'
```

We are going to predict the likelihood of a bite incident occuring in a dog's lifetime based on size, breed, and age.

We will compare this data by people's perception based on breed banning in different states.

Conclusion

Our goal is to help people understand that all dogs need to be trained regardless of breed.
We hope to give knowledge to everyone to help find the breeds that are suitable for them to prevent people from giving up their dogs.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: