Data Visualization and Analytics of New York Accidents from 2016 to 2021

Brief Introduction

A data analysis of the one of the busiest cities in North America and even in the world

Table of Contents

1. Overview of the Data Analytics and Visualization

This will cover cover the following visualizations of the dataset:

- The top 15 cities and counties in the state of New York with the most accidents

2. Factors in the Analysis

- City
- County

3. Packages Needed

Below are the pre-requisites for processing the dataset:

IMPORTS

In [1]:
```
# Packages that needs to be installed

#pip install pandas
#pip install seaborn
#pip install pyarrow
#conda install geopandas
#pip install datashader
#pip install "holoviews[recommended]"
#pip install hvplot
```

In [1]:
```
# Imports for data processing
import pandas as pd
import numpy as np
```

In [2]:
```
# Imports for descriptive analytics and for correlational analytics

# Matplotlib imports
import matplotlib.pyplot as plt

# Seaborn imports
import seaborn as sns

sns.set_theme(style="whitegrid")
```

In [3]:
```
# Imports for map analyatics
import geopandas as gpd

# Datashader imports
import datashader as ds
import datashader.transfer_functions as tf
from datashader.utils import export_image
from datashader.utils import lnglat_to_meters as webm
from datashader.colors import colormap_select

# Holoviews imports
import holoviews as hv

# Utility imports
from functools import partial
from colorcet import fire
from IPython.display import HTML, display

hv.extension('bokeh', 'matplotlib')

%matplotlib inline
```

3.1 Information About The Dataset

Link for all dataset files

> Google Drive Link: https://drive.google.com/drive/folders/1EeLJvzItBig7DdLTiZ2_2siO1N2mifQ7?usp=share_link

We only need the following columns for the data analysis of this dataset:

Details from kaggle: https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents

Severity - how severe the accident where in it ranges from 1 to 4 where 1 indicates the least impact on traffic
Start_Time - time when the accident initially occured
End_Lat - latitude in the gps
End_Lng - longitude in the gps
City - address of city
County - address of county
State - address of the state
Country - address country
Temperature - temperature that was present (in Fahrenheit)
Visibility - visibility at that time (in miles)
Wind_Speed - the wind speed in mph
Precipitation - amount of rainfall in inches
Weather_Condition - condition of wether at that time e.g. rain, thunderstorm, blizzard

4. Data Visualization

In [4]:
```
# Reads the final processed version

us_accidents_dataset_final = pd.read_parquet("us_accidents_final.parq")
us_accidents_dataset_final.head()
```

Out[4]:

	Severity	Start_Time	End_Lat	End_Lng	Side	City	County	State	Country	Temperature(F)	Visibility(mi)	W
0	3	2016-02-08 00:37:08	40.112061	-83.031868	R	Dublin	Franklin	OH	US	42.099998	10.0	
1	2	2016-02-08 05:56:20	39.865009	-84.048729	R	Dayton	Montgomery	OH	US	36.900002	10.0	
2	2	2016-02-08 06:15:39	39.102089	-84.523956	R	Cincinnati	Hamilton	OH	US	36.000000	10.0	
3	2	2016-02-08 06:51:45	41.062168	-81.535469	R	Akron	Summit	OH	US	39.000000	10.0	
4	3	2016-02-08 07:53:43	39.170475	-84.501801	R	Cincinnati	Hamilton	OH	US	37.000000	10.0	

Important Note:

> When you run the plots and all of them are in stacked with each other, just run the idividual plot again

4.1 Descriptive Analytics

> ## HELPER FUNCTIONS

In [5]:
```python
# HELPER FUNCTIONS

def group_aggregate(dataset, group_by_column, agg_columns: dict):
    """This is used to groupby a dataset then also aggregated the given columns

    Args:
        dataset (dataframe): the dataset to be passed
        group_by_column (column): the column to be group by
        agg_columns (dict): the column(s) to be aggegrated

    Returns:
        dataset: returns the grouped and aggregated dataset
    """

    dataset = dataset.copy()
    aggregated_dataset = dataset.groupby([group_by_column]).agg(agg_columns).reset_index()

    return aggregated_dataset


def get_n_rows_sorted(dataset, n_rows, column):
    """Gets the largest values in the dataset then sorts it in a descending order

    Args:
        dataset (dataframe): the dataset to be passed
        n_rows (int): the number of rows to be sorted
        column (int): the column to be sorted

    Returns:
        dataset: returns a dataset that is sorted in a descending order with the largest values
    """

    return dataset.nlargest(n_rows, column, keep='last')

def round_off(dataset, column, sig_figures):
    """This rounds of the values of a column that contains float type then returns a float value of it

    Args:
        dataset (dataframe): the dataset to be passed
```

```
        column (float): the column to be rounded off (must be a float)
        sig_figures (int): the number of significant figures for rounding off

    Returns:
        column: returns the rounded off column
    """

    return dataset.round({column: sig_figures})

def plot_barplot(dataset, x_range, y_range, order, color, orient, hue=None):
    """Plots a bar plot

    Args:
        dataset (dataframe): dataset to be oassed
        x_range (any): the x_range of the plot
        y_range (any): the y_range of the plot
        order (any): how the values will be arranged
        color (str): pallete of the plot
        orient (str): vertical or horizontal (v or h)
    """

    sns.barplot(data=dataset, x=x_range, y=y_range, order=dataset[order], palette=color, orient=orient, h
```

In [36]:
```
# Addes the accidents for each state then puts on a new column called Total_Accidents
us_accidents_total = us_accidents_dataset_final.copy()
us_accidents_total["Total_Accidents"] = us_accidents_total["State"].map(us_accidents_total["State"].value
```

In [37]:
```
us_accidents_total.head()
```

Out[37]:

	Severity	Start_Time	End_Lat	End_Lng	Side	City	County	State	Country	Temperature(F)	Visibility(mi)	W
0	3	2016-02-08 00:37:08	40.112061	-83.031868	R	Dublin	Franklin	OH	US	42.099998	10.0	
1	2	2016-02-08 05:56:20	39.865009	-84.048729	R	Dayton	Montgomery	OH	US	36.900002	10.0	
2	2	2016-02-08 06:15:39	39.102089	-84.523956	R	Cincinnati	Hamilton	OH	US	36.000000	10.0	
3	2	2016-02-08 06:51:45	41.062168	-81.535469	R	Akron	Summit	OH	US	39.000000	10.0	
4	3	2016-02-08 07:53:43	39.170475	-84.501801	R	Cincinnati	Hamilton	OH	US	37.000000	10.0	

In [45]:
```
nyc_accidents_data = us_accidents_total.copy()
nyc_accidents_data.head()
```

Out[45]:

	Severity	Start_Time	End_Lat	End_Lng	Side	City	County	State	Country	Temperature(F)	Visibility(mi)	W
0	3	2016-02-08 00:37:08	40.112061	-83.031868	R	Dublin	Franklin	OH	US	42.099998	10.0	
1	2	2016-02-08 05:56:20	39.865009	-84.048729	R	Dayton	Montgomery	OH	US	36.900002	10.0	
2	2	2016-02-08 06:15:39	39.102089	-84.523956	R	Cincinnati	Hamilton	OH	US	36.000000	10.0	
3	2	2016-02-08 06:51:45	41.062168	-81.535469	R	Akron	Summit	OH	US	39.000000	10.0	
4	3	2016-02-08 07:53:43	39.170475	-84.501801	R	Cincinnati	Hamilton	OH	US	37.000000	10.0	

In [46]:
```
nyc_accidents_data.set_index("State", inplace = True)
nyc_accidents_data = nyc_accidents_data.loc["NY"]
nyc_accidents_data.head()
```

Out[46]:

State	Severity	Start_Time	End_Lat	End_Lng	Side	City	County	Country	Temperature(F)	Visibility(mi)	Wind_S
NY	2	2016-11-30 15:58:59	40.853291	-73.960640	R	New York	New York	US	53.099998	2.0	
NY	2	2016-11-30 17:13:53	41.129971	-74.167732	R	Central Valley	Orange	US	51.799999	3.0	
NY	2	2016-11-30 17:12:20	43.042728	-76.142441	R	Syracuse	Onondaga	US	53.099998	10.0	
NY	2	2016-11-30 17:31:55	40.768780	-73.949059	R	New York	New York	US	53.099998	2.5	
NY	2	2016-11-30 17:31:55	40.699459	-73.984154	R	Brooklyn	Kings	US	53.599998	NaN	

4.1a Total Accidents Per Year in the State of New York From 2016 to 2021

In [47]:
```
# Copies the us_accidents_total data set into a new variable called us_accidents_timespan
nyc_accidents_timespan = nyc_accidents_data.copy()
```

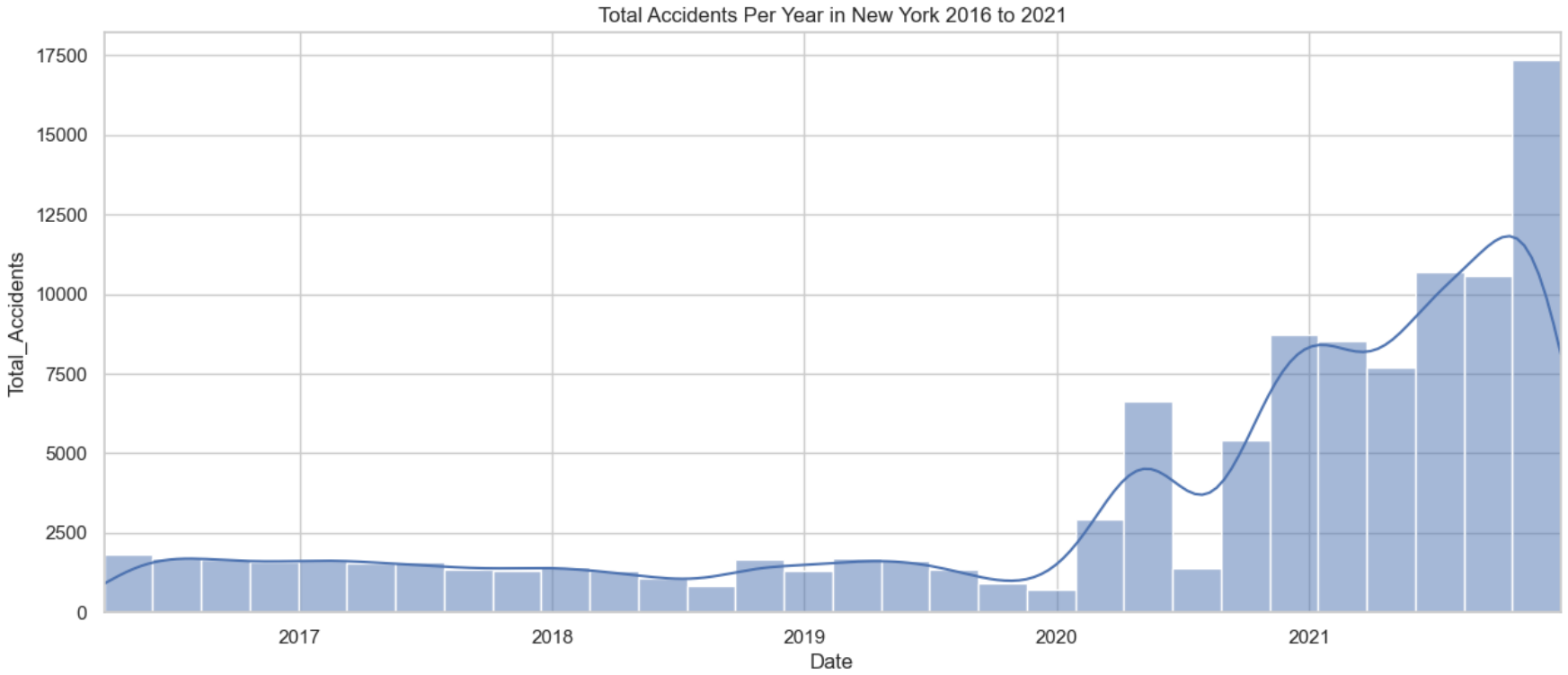
In [48]:
```
# Plotting histogram of the accidents per year from 2016 to 2021 in the US

f, ax = plt.subplots(figsize=(15, 6))
plt.ticklabel_format(style='plain', axis='y')

fig = sns.histplot(data=nyc_accidents_data, x='Date', bins=30, kde=True)
fig.set_xlim(nyc_accidents_timespan['Date'].min(), nyc_accidents_timespan['Date'].max())
fig.set_ylabel("Total_Accidents")

plt.title("Total Accidents Per Year in New York 2016 to 2021")
```

Out[48]:
```
Text(0.5, 1.0, 'Total Accidents Per Year in New York 2016 to 2021')
```



4.1b Top 15 Cities in the State of New York With the Most Accidents

In [49]:
```
# The aggregrated top 15 states and their total accidents and the severity for each state

nyc_accidents_top15_cities = group_aggregate(dataset=nyc_accidents_data, group_by_column='City', agg_colu
nyc_accidents_top15_cities = get_n_rows_sorted(dataset=nyc_accidents_top15_cities, n_rows=15, column='Tot
nyc_accidents_top15_cities = round_off(dataset=nyc_accidents_top15_cities, column='Severity', sig_figures
nyc_accidents_top15_cities.head(15)
```

Out[49]:

	City	Total_Accidents
110	Bronx	9453
795	Rochester	8224
654	New York	7068
113	Brooklyn	6035
116	Buffalo	3361
924	Syracuse	2277
8	Albany	2197
324	Flushing	1675
909	Staten Island	1561
473	Jamaica	1293
1048	Yonkers	1263
842	Schenectady	985
819	Rye	915
574	Maspeth	914
218	Corona	908

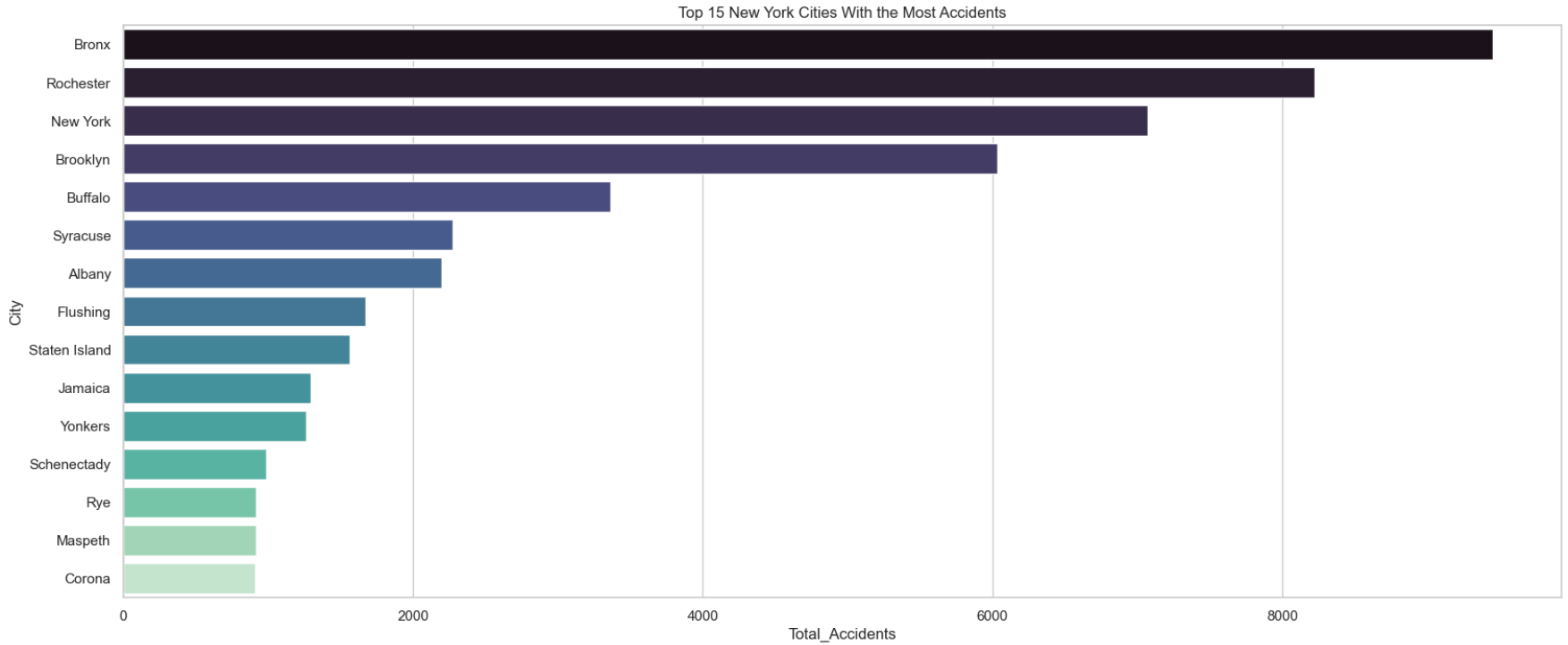
In [50]:
```
f, ax = plt.subplots(figsize=(20, 8))

plt.ticklabel_format(style='plain', axis='x')

plot_barplot(dataset=nyc_accidents_top15_cities, x_range='Total_Accidents', y_range='City', order='City',

plt.title("Top 15 New York Cities With the Most Accidents")
```

Out[50]:
```
Text(0.5, 1.0, 'Top 15 New York Cities With the Most Accidents')
```



4.1b Top 15 Counties in the State of New York With the Most Accidents

In [51]:
```
# The aggregrated top 15 states and their total accidents and the severity for each state

nyc_accidents_top15_counties = group_aggregate(dataset=nyc_accidents_data, group_by_column='County', agg_
nyc_accidents_top15_counties = get_n_rows_sorted(dataset=nyc_accidents_top15_counties, n_rows=15, column=
nyc_accidents_top15_counties = round_off(dataset=nyc_accidents_top15_counties, column='Severity', sig_fig
nyc_accidents_top15_counties.head(15)
```

Out[51]:

	County	Total_Accidents
40	Queens	12556
27	Monroe	10594
60	Westchester	9913
2	Bronx	9461
30	New York	7155
29	Nassau	6219
23	Kings	5983
14	Erie	4693
0	Albany	4488
52	Suffolk	4467
33	Onondaga	3782
32	Oneida	2950
45	Saratoga	2907
43	Rockland	2781
13	Dutchess	2166

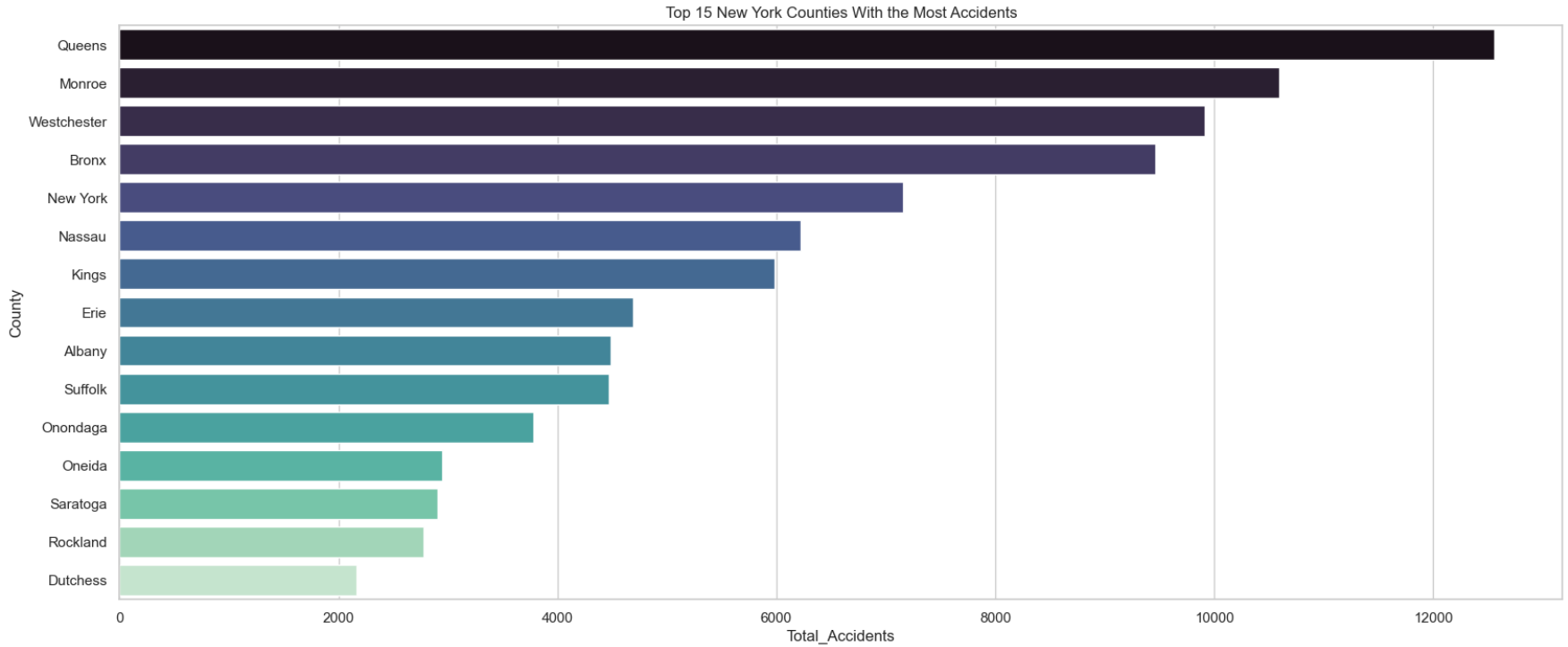
In [52]:
```
f, ax = plt.subplots(figsize=(20, 8))

plt.ticklabel_format(style='plain', axis='x')

plot_barplot(dataset=nyc_accidents_top15_counties, x_range='Total_Accidents', y_range='County', order='Co

plt.title("Top 15 New York Counties With the Most Accidents")
```

Out[52]:
```
Text(0.5, 1.0, 'Top 15 New York Counties With the Most Accidents')
```



4.2 Map Analytics

This section will focus on the aggregated accidents account of all states and heatmaps

4.2a Choropleth Map of the New York Accidents by County

In [53]:
```
nyc_accidents_counties = group_aggregate(dataset=nyc_accidents_data, group_by_column='County', agg_columr
nyc_accidents_counties = get_n_rows_sorted(dataset=nyc_accidents_counties, n_rows=100, column='Total_Acci
nyc_accidents_counties = round_off(dataset=nyc_accidents_counties, column='Severity', sig_figures=1)
nyc_accidents_counties.head(62)
```

Out[53]:

	County	Total_Accidents
40	Queens	12556
27	Monroe	10594
60	Westchester	9913
2	Bronx	9461
30	New York	7155
...	...	...
59	Wayne	27
20	Hamilton	20
1	Allegany	18
48	Schuyler	15
62	Yates	11

62 rows × 2 columns

In [54]:
```
# Getting the shape file for the map
shape_path = './NY-Counties/Counties.shp'
nyc_map_shp = gpd.read_file(shape_path)
```

In [55]:
```
nyc_map_merged = nyc_map_shp.set_index('NAME').join(nyc_accidents_counties.set_index('County'))
nyc_map_merged.head()
```

Out[55]:

NAME	ABBREV	GNIS_ID	FIPS_CODE	SWIS	NYSP_ZONE	POP1990	POP2000	POP2010	POP2020	DOS_LL	DOSLL_
Albany	ALBA	974099	36001	010000	East	292594	294565	304204	314848	NaN	
Allegany	ALLE	974100	36003	020000	West	50470	49927	48946	46456	NaN	
Bronx	BRON	974101	36005	600000	Long Island	1203789	1332650	1385108	1472654	NaN	
Broome	BROO	974102	36007	030000	Central	212160	200536	200600	198683	NaN	
Cattaraugus	CATT	974103	36009	040000	West	84234	83955	80317	77042	NaN	

In [56]:

```
# Merging the shape file with the total accidents per county
nyc_map = nyc_map_shp.merge(nyc_accidents_counties[['County', 'Total_Accidents']], left_on='NAME', right_
nyc_map.head()
```

Out[56]:

	NAME	ABBREV	GNIS_ID	FIPS_CODE	SWIS	NYSP_ZONE	POP1990	POP2000	POP2010	POP2020	DOS_LL	DOSL
0	Albany	ALBA	974099	36001	010000	East	292594	294565	304204	314848	NaN	
1	Allegany	ALLE	974100	36003	020000	West	50470	49927	48946	46456	NaN	
2	Bronx	BRON	974101	36005	600000	Long Island	1203789	1332650	1385108	1472654	NaN	
3	Broome	BROO	974102	36007	030000	Central	212160	200536	200600	198683	NaN	
4	Cattaraugus	CATT	974103	36009	040000	West	84234	83955	80317	77042	NaN	

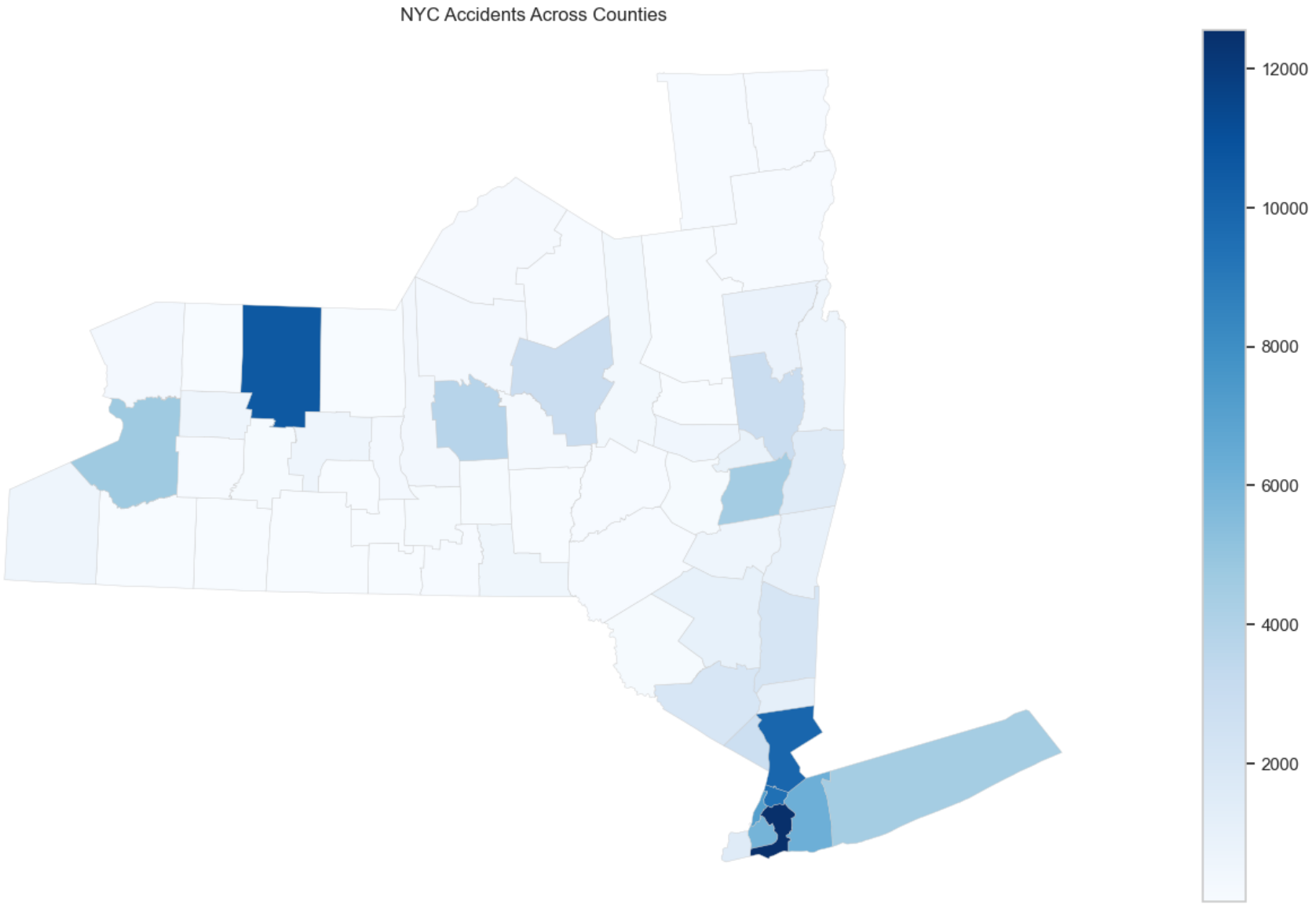
In [57]:
```
fig, ax = plt.subplots(1, figsize=(20, 10))

variable = 'Total_Accidents'
vmin, vmax = 120, 1000

# add a title
ax.set_title('NYC Accidents Across Counties', fontdict={'fontsize': '12', 'fontweight' : '3'})

nyc_map_merged.plot(column=variable, cmap='Blues', linewidth=0.3, ax=ax, edgecolor='0.8', legend=True)
ax.axis('off')
```

Out[57]:
```
(71853.38954500001, 813650.0947549996, 4455716.4048, 5010703.0898)
```


NYC Accidents Across Counties
12000
10000
8000
6000
4000
2000

Since it is very difficult to implement labels in the patches themselves, we will be base from the image below for the counties of New York

Link for the county reference: https://drive.google.com/file/d/177JvVqFOrjTAFzbnpVn6qbd9DbgH90iV/view?usp=share_link

4.2b New York Accidents Heatmap

In [58]:
```
# Setting the width and height of the heatmap
plot_width  = int(3840)
plot_height = int(2160)
```

In [59]:
```
# Setting the heatmap colormaps and background

background = "black"

export = partial(export_image, background=background, export_path="export")
cm = partial(colormap_select, reverse=(background!="black"))

display(HTML("<style>.container  { width:100% !important; }</style>"))
```

In [60]:
```
def render_heatmap(dataset, x_range, y_range, file_name, location = None):
    """This function is created to reduce code duplication and to also take in
       arguments in one line. The main goal of this function to render a heatmap
       the chosen dataset with the following paramters:

    Args:
        dataset (dataframe): dataset or dataframe
        x_range (float): x_axis in the mercator format
        y_range (float): y_axis in the mercator format
        location (tuple): the boundaries of the location

    Returns:
        image: returns a render of the heatmap and exports it locally in a 'export' folder
    """

    # Checks if a location parameter is passed if there is
    # then will return it with converted web mercator format,
    # if none then it will not pass in the location into the canvas
    #
    # note: the location must be tuples of tuples

    if location is not None:
        cvs = ds.Canvas(plot_width, plot_height, *webm(*location))
    elif location is None:
        cvs = ds.Canvas(plot_width, plot_height)

    agg = cvs.points(dataset, x_range, y_range)

    return export(tf.shade(agg, cmap=cm(fire), how='eq_hist'), file_name)
```

In [61]:
```
# Helper function for finding the state boundaries and the coordinates of their accidents

def get_per_state_accidents(dataset, state_id):

    return dataset.loc[dataset['State'] == state_id]
```

In [62]:
```
us_accidents_total.head()
```

Out[62]:

	Severity	Start_Time	End_Lat	End_Lng	Side	City	County	State	Country	Temperature(F)	Visibility(mi)	W
0	3	2016-02-08 00:37:08	40.112061	-83.031868	R	Dublin	Franklin	OH	US	42.099998	10.0	
1	2	2016-02-08 05:56:20	39.865009	-84.048729	R	Dayton	Montgomery	OH	US	36.900002	10.0	
2	2	2016-02-08 06:15:39	39.102089	-84.523956	R	Cincinnati	Hamilton	OH	US	36.000000	10.0	
3	2	2016-02-08 06:51:45	41.062168	-81.535469	R	Akron	Summit	OH	US	39.000000	10.0	
4	3	2016-02-08 07:53:43	39.170475	-84.501801	R	Cincinnati	Hamilton	OH	US	37.000000	10.0	

In [63]:
```
# Getting the accidents of New York along with their easting and northing values
us_states_heatmap = us_accidents_total.copy()
us_states_heatmap = us_states_heatmap[['State', 'Easting', 'Northing']]

NEW_YORK = get_per_state_accidents(us_states_heatmap, state_id='NY')
```

In [67]:
```
# Create the heatmaps

new_york_heatmap = render_heatmap(dataset=NEW_YORK, x_range='Easting', y_range='Northing', file_name="New
```

In [66]:
```
new_york_heatmap
```

Out[66]: