US Accidents (2016-2020)

The United States is one of the busiest countries in terms of road traffic with nearly 284 million vehicles in operation as of the third quarter of 2021. It was further reported that more than 228.7 million drivers were holding a valid driving license as of 2019. This level of traffic is one of the reasons leading to more road accidents: In 2019, there were some 12.15 million vehicles involved in crashes in the United States, with over half that volume being passenger cars. The number of road accidents per one million inhabitants in the U.S. is forecast to dip down in the next years, reaching just over 7,100 in 2025. A slower decrease is further tipped for the number of fatalities due to traffic accidents.

Injuries and fatalities on the road

The United States is among the countries with the highest rate of traffic-related fatalities per one million population. The rate of traffic fatalities per 100 million annual vehicle miles travelled (VMT) peaked in 2016, before slowly dropping down to 1.11 traffic-related fatalities per 100 million VTM in 2019. While traffic-related fatalities were lower in 2019, they spiked up to the highest level recorded since 2016 in 2020. This was most visible as the country reopened following stay-at-home orders due to the COVID-19 pandemic. All States were partially reopened by June 2020, month were fatalities started to increase year-over-year. Most of the fatalities recorded were occupants of vehicles, excluding motorcyclists.

In non-fatal crashes, drivers and motor vehicle passengers also recorded the most traffic-related injuries. This was in part due to the mass motorization in the United States, which recorded a higher number of cars and light trucks on the road when compared to other private modes of transport. The total number of injured people in motor vehicle crashes reported a sharp increase between 2015 and 2016, before dipping in 2017 and flattening through 2019.

Evolution and mitigation of the risk factors

Drunk driving and speeding were two of the main road accident risk factors in the United States. In 2019, speeding-related accidents accounted for close to 9,500 traffic fatalities, while alcohol-impaired driving fatalities represented over a quarter of the deaths recorded that same year. 21 to 24-year-olds were the age group most at risk of being involved in an alcohol-impaired driving fatal crash, with over a quarter of the age group involved in these types of accidents in 2019. All 50 states have a .08 blood alcohol concentration limit for drivers, after which they are driving under the influence of alcohol.

Risk factor mitigation is one of the main concerns of the population. In a 2021 survey, U.S. consumers highlighted car safety as the most important characteristic when deciding on a new car, above fuel efficiency, high quality, and low prices. Vehicle seat belts and motorcycle helmets are some of the safety tools helping mitigate the risks on the roads. The U.S. recorded a stable seatbelt use, at 90.3 percent in 2020, but motorcycle helmet use dropped under 70 percent in that same year. State legislations vary regarding helmet use. In states requiring motorcyclists to wear one, a higher usage rate of helmets compliant with the Department of Transportation’s guidelines was recorded. These legislations also almost doubled the share of non-compliant helmet use when compared to other states, and the share of motorcyclists wearing no helmet shrunk from over 40 percent to under six percent in states with helmet requirements

US Accident Data Analysis

# Data Downloading

In [37]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
!pip install Pyppeteer
```

```
Collecting Pyppeteer
  Downloading pyppeteer-1.0.2-py3-none-any.whl (83 kB)
Requirement already satisfied: certifi>=2021 in c:\users\vitrogene\anaconda3\lib\site-pack
ages (from Pyppeteer) (2021.10.8)
Collecting websockets<11.0,>=10.0
  Downloading websockets-10.1-cp39-cp39-win_amd64.whl (97 kB)
Requirement already satisfied: importlib-metadata>=1.4 in c:\users\vitrogene\anaconda3\lib
\site-packages (from Pyppeteer) (4.8.1)
Requirement already satisfied: appdirs<2.0.0,>=1.4.3 in c:\users\vitrogene\anaconda3\lib\s
ite-packages (from Pyppeteer) (1.4.4)
Requirement already satisfied: urllib3<2.0.0,>=1.25.8 in c:\users\vitrogene\anaconda3\lib
\site-packages (from Pyppeteer) (1.26.7)
Requirement already satisfied: tqdm<5.0.0,>=4.42.1 in c:\users\vitrogene\anaconda3\lib\sit
e-packages (from Pyppeteer) (4.62.3)
Collecting pyee<9.0.0,>=8.1.0
  Downloading pyee-8.2.2-py2.py3-none-any.whl (12 kB)
Requirement already satisfied: zipp>=0.5 in c:\users\vitrogene\anaconda3\lib\site-packages
(from importlib-metadata>=1.4->Pyppeteer) (3.6.0)
Requirement already satisfied: colorama in c:\users\vitrogene\anaconda3\lib\site-packages
(from tqdm<5.0.0,>=4.42.1->Pyppeteer) (0.4.4)
Installing collected packages: websockets, pyee, Pyppeteer
Successfully installed Pyppeteer-1.0.2 pyee-8.2.2 websockets-10.1
```

In [3]:

```
df=pd.read_csv('US_Accidents_Dec20_updated.csv')
df
```

Out[3]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Dist
0	A-2716600	3	2016-02-08 00:37:08	2016-02-08 06:37:08	40.10891	-83.09286	40.11206	-83.03187	
1	A-2716601	2	2016-02-08 05:56:20	2016-02-08 11:56:20	39.86542	-84.06280	39.86501	-84.04873	
2	A-2716602	2	2016-02-08 06:15:39	2016-02-08 12:15:39	39.10266	-84.52468	39.10209	-84.52396	
3	A-2716603	2	2016-02-08 06:15:39	2016-02-08 12:15:39	39.10148	-84.52341	39.09841	-84.52241	
4	A-2716604	2	2016-02-08 06:51:45	2016-02-08 12:51:45	41.06213	-81.53784	41.06217	-81.53547	
...	...	...	...	...	...	...	...	...	

Loading [MathJax]/extensions/Safe.js

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Dist
1516059	A-4239402	2	2019-08-23 18:03:25	2019-08-23 18:32:01	34.00248	-117.37936	33.99888	-117.37094	
1516060	A-4239403	2	2019-08-23 19:11:30	2019-08-23 19:38:23	32.76696	-117.14806	32.76555	-117.15363	
1516061	A-4239404	2	2019-08-23 19:00:21	2019-08-23 19:28:49	33.77545	-117.84779	33.77740	-117.85727	
1516062	A-4239405	2	2019-08-23 19:00:21	2019-08-23 19:29:42	33.99246	-118.40302	33.98311	-118.39565	
1516063	A-4239406	2	2019-08-23 18:52:06	2019-08-23 19:21:31	34.13393	-117.23092	34.13736	-117.23934	

1516064 rows × 47 columns

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1516064 entries, 0 to 1516063
Data columns (total 47 columns):
 #   Column                 Non-Null Count    Dtype  
---  ------                 --------------    -----  
 0   ID                     1516064 non-null  object 
 1   Severity               1516064 non-null  int64  
 2   Start_Time             1516064 non-null  object 
 3   End_Time               1516064 non-null  object 
 4   Start_Lat              1516064 non-null  float64
 5   Start_Lng              1516064 non-null  float64
 6   End_Lat                1516064 non-null  float64
 7   End_Lng                1516064 non-null  float64
 8   Distance(mi)           1516064 non-null  float64
 9   Description            1516064 non-null  object 
 10  Number                 469969 non-null   float64
 11  Street                 1516064 non-null  object 
 12  Side                   1516064 non-null  object 
 13  City                   1515981 non-null  object 
 14  County                 1516064 non-null  object 
 15  State                  1516064 non-null  object 
 16  Zipcode                1515129 non-null  object 
 17  Country                1516064 non-null  object 
 18  Timezone               1513762 non-null  object 
 19  Airport_Code           1511816 non-null  object 
 20  Weather_Timestamp      1485800 non-null  object 
 21  Temperature(F)         1473031 non-null  float64
 22  Wind_Chill(F)          1066748 non-null  float64
 23  Humidity(%)            1470555 non-null  float64
 24  Pressure(in)           1479790 non-null  float64
 25  Visibility(mi)         1471853 non-null  float64
 26  Wind_Direction         1474206 non-null  object 
 27  Wind_Speed(mph)        1387202 non-null  float64
 28  Precipitation(in)      1005515 non-null  float64
 29  Weather_Condition      1472057 non-null  object 
 30  Amenity                1516064 non-null  bool   
                            1516064 non-null  bool   
```

Loading [MathJax]/extensions/Safe.js

```
 32  Crossing               1516064 non-null  bool
 33  Give_Way               1516064 non-null  bool
 34  Junction               1516064 non-null  bool
 35  No_Exit                1516064 non-null  bool
 36  Railway                1516064 non-null  bool
 37  Roundabout             1516064 non-null  bool
 38  Station                1516064 non-null  bool
 39  Stop                   1516064 non-null  bool
 40  Traffic_Calming        1516064 non-null  bool
 41  Traffic_Signal         1516064 non-null  bool
 42  Turning_Loop           1516064 non-null  bool
 43  Sunrise_Sunset         1515981 non-null  object
 44  Civil_Twilight         1515981 non-null  object
 45  Nautical_Twilight      1515981 non-null  object
 46  Astronomical_Twilight  1515981 non-null  object
dtypes: bool(13), float64(13), int64(1), object(20)
memory usage: 412.1+ MB
```

ask question and answer 1.are there more accident in warmer or colder areas? 2.which states has the number of accidents? -----

In [5]:

```
df.describe()
```

Out[5]:

	Severity	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(mi)	
count	1.516064e+06	1.516064e+06	1.516064e+06	1.516064e+06	1.516064e+06	1.516064e+06	4.699
mean	2.238630e+00	3.690056e+01	-9.859919e+01	3.690061e+01	-9.859901e+01	5.872617e-01	8.907
std	6.081481e-01	5.165653e+00	1.849602e+01	5.165629e+00	1.849590e+01	1.632659e+00	2.242
min	1.000000e+00	2.457022e+01	-1.244976e+02	2.457011e+01	-1.244978e+02	0.000000e+00	0.000
25%	2.000000e+00	3.385422e+01	-1.182076e+02	3.385420e+01	-1.182077e+02	0.000000e+00	1.212
50%	2.000000e+00	3.735113e+01	-9.438100e+01	3.735134e+01	-9.437987e+01	1.780000e-01	4.000
75%	2.000000e+00	4.072593e+01	-8.087469e+01	4.072593e+01	-8.087449e+01	5.940000e-01	1.010
max	4.000000e+00	4.900058e+01	-6.711317e+01	4.907500e+01	-6.710924e+01	1.551860e+02	9.999

In [6]:

```
Missing_percentage=df.isna().sum().sort_values(ascending=False)/len(df)
Missing_percentage
```

Out[6]:

```
Number                   0.690007
Precipitation(in)        0.336760
Wind_Chill(F)            0.296370
Wind_Speed(mph)          0.084998
Humidity(%)              0.030018
Visibility(mi)           0.029162
Weather_Condition        0.029027
Temperature(F)           0.028385
Wind_Direction           0.027610
Pressure(in)             0.023926
Weather_Timestamp        0.019962
Airport_Code             0.002802
Timezone                 0.001518
Zipcode                  0.000617
Sunrise_Sunset           0.000055
Civil_Twilight           0.000055
Nautical_Twilight        0.000055
Astronomical_Twilight    0.000055
City                     0.000055
Country                  0.000000
Give_Way                 0.000000
                         0.000000
```

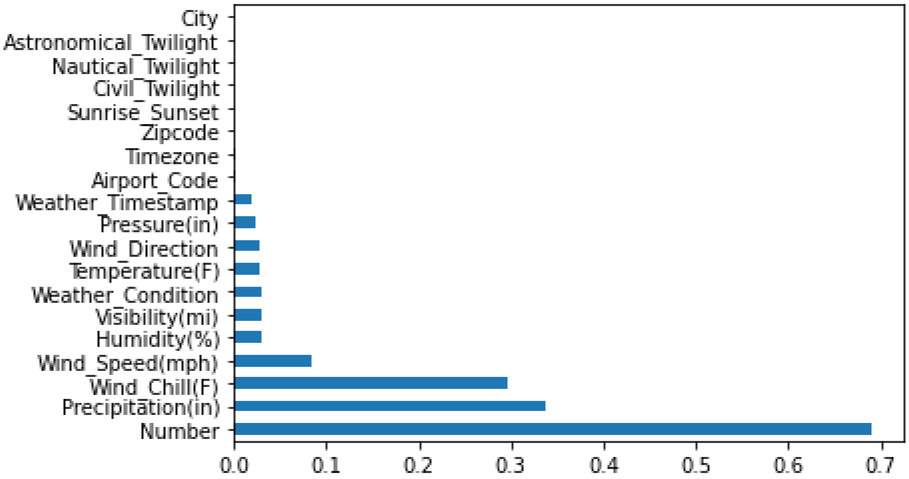
```
End_Time                 0.000000
Start_Lat                0.000000
Turning_Loop             0.000000
Traffic_Signal           0.000000
Traffic_Calming          0.000000
Stop                     0.000000
Station                  0.000000
Roundabout               0.000000
Railway                  0.000000
No_Exit                  0.000000
Junction                 0.000000
Crossing                 0.000000
State                    0.000000
Bump                     0.000000
Amenity                  0.000000
Start_Lng                0.000000
End_Lat                  0.000000
End_Lng                  0.000000
Distance(mi)             0.000000
Description              0.000000
Street                   0.000000
Severity                 0.000000
Side                     0.000000
County                   0.000000
ID                       0.000000
dtype: float64
```

In [7]:

```
Missing_percentage[Missing_percentage !=0.].plot(kind='barh')
```

Out[7]:

```
<AxesSubplot:>
```



the columns we do not need

In [8]:

```
df_delete = df.drop(['Number'], axis=1)
```

In [9]:

```
Cities_by_accident = df.City.value_counts().sort_values(ascending=False)
Cities_by_accident
```

Out[9]:

```
Los Angeles                     39984
Miami                           36233
Charlotte                       22203
Houston                         20843
Dallas                          19497
                                ...  
Holmen                              1
                                    1
```

Loading [MathJax]/extensions/Safe.js

```
Downing                          1
Glenwood City                    1
American Fork-Pleasant Grove     1
Name: City, Length: 10657, dtype: int64
```

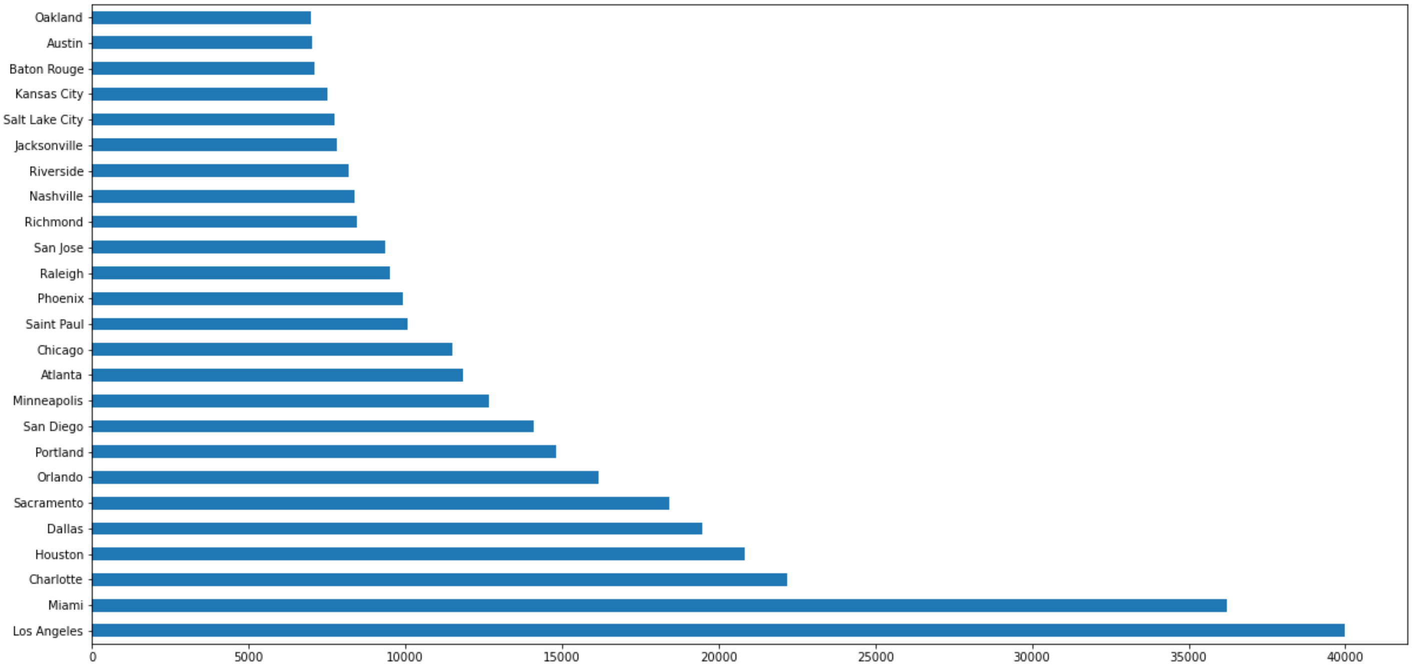
Top 25 Cities

In [10]:

```
Cities_by_accident[:25].plot(kind='barh',figsize=(20,10))
```

Out[10]:

```
<AxesSubplot:>
```



In [11]:

```
cities =df.City.unique()
len(cities)
```

Out[11]:

```
10658
```

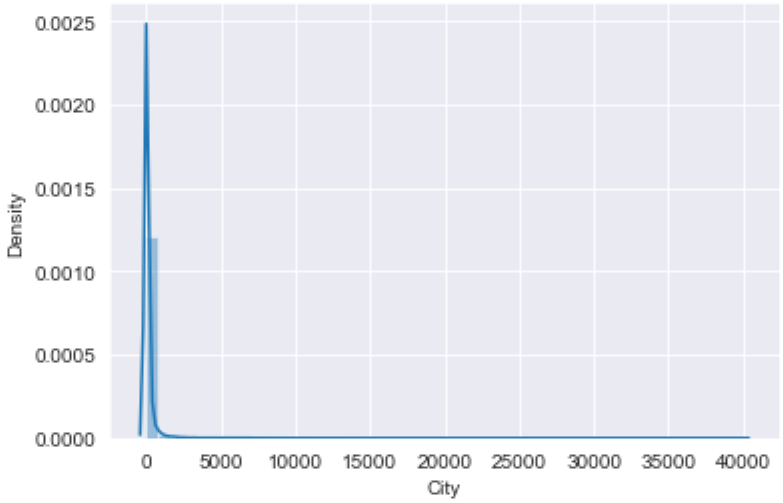
In [12]:

```
sns.set_style("darkgrid")
sns.distplot(Cities_by_accident)
```

```
C:\Users\Vitrogene\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please ada
pt your code to use either `displot` (a figure-level function with similar flexibility) or
`histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[12]:

```
<AxesSubplot:xlabel='City', ylabel='Density'>
```



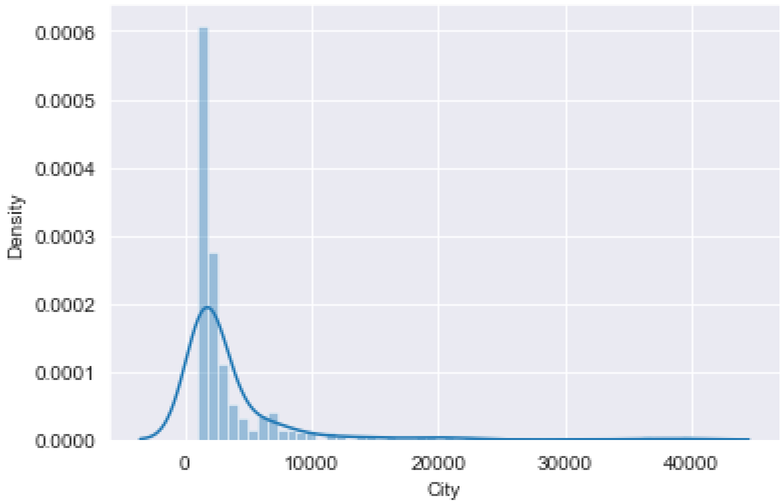
```
In [13]:  high_accident_cities = Cities_by_accident[Cities_by_accident >= 1000]
          low_accident_cities = Cities_by_accident[Cities_by_accident < 1000]
```

```
In [14]:  len(high_accident_cities)/len(cities)

Out[14]:  0.023550384687558643
```

```
In [15]:  sns.distplot(high_accident_cities,)
```

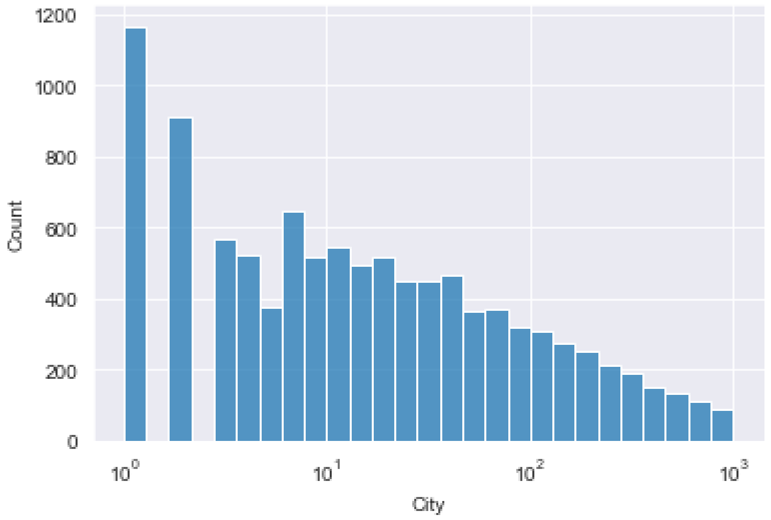
```
C:\Users\Vitrogene\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please ada
pt your code to use either `displot` (a figure-level function with similar flexibility) or
`histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
Out[15]:  <AxesSubplot:xlabel='City', ylabel='Density'>
```



```
In [16]:  sns.histplot(low_accident_cities, log_scale=True)

Out[16]:  <AxesSubplot:xlabel='City', ylabel='Count'>
```

Loading [MathJax]/extensions/Safe.js



# Start Time

In [17]:

```
df.Start_Time
```

Out[17]:

```
0          2016-02-08 00:37:08
1          2016-02-08 05:56:20
2          2016-02-08 06:15:39
3          2016-02-08 06:15:39
4          2016-02-08 06:51:45
                  ...
1516059    2019-08-23 18:03:25
1516060    2019-08-23 19:11:30
1516061    2019-08-23 19:00:21
1516062    2019-08-23 19:00:21
1516063    2019-08-23 18:52:06
Name: Start_Time, Length: 1516064, dtype: object
```

In [18]:

```
df.Start_Time = pd.to_datetime(df.Start_Time)
df.Start_Time
```

Out[18]:

```
0         2016-02-08 00:37:08
1         2016-02-08 05:56:20
2         2016-02-08 06:15:39
3         2016-02-08 06:15:39
4         2016-02-08 06:51:45
                  ...
1516059   2019-08-23 18:03:25
1516060   2019-08-23 19:11:30
1516061   2019-08-23 19:00:21
1516062   2019-08-23 19:00:21
1516063   2019-08-23 18:52:06
Name: Start_Time, Length: 1516064, dtype: datetime64[ns]
```

Hour basis accidents

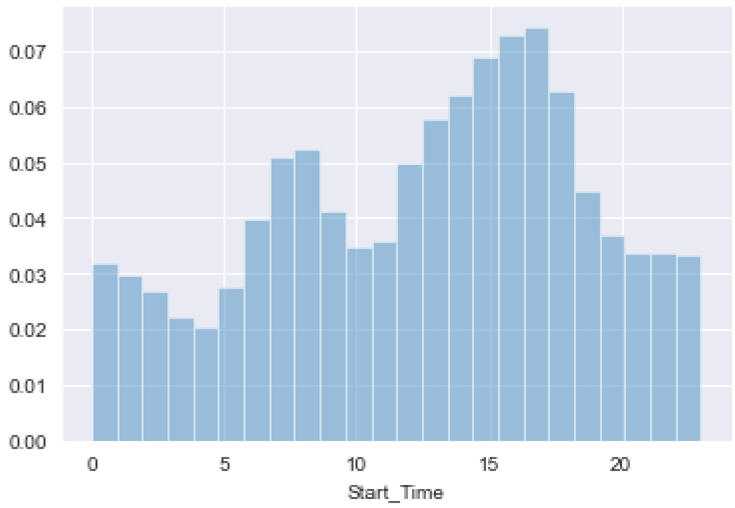
In [19]:

```
sns.distplot(df.Start_Time.dt.hour, bins=24, kde=False, norm_hist=True)
```

```
C:\Users\Vitrogene\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please ada
pt your code to use either `displot` (a figure-level function with similar flexibility) or
`histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[19]:

```
<AxesSubplot:xlabel='Start_Time'>
```



- High percentage of accidents occur 1PM to 6PM.
- Probably people are in hurry or due to traffic
- Next highest percentage is 6AM to 9AM

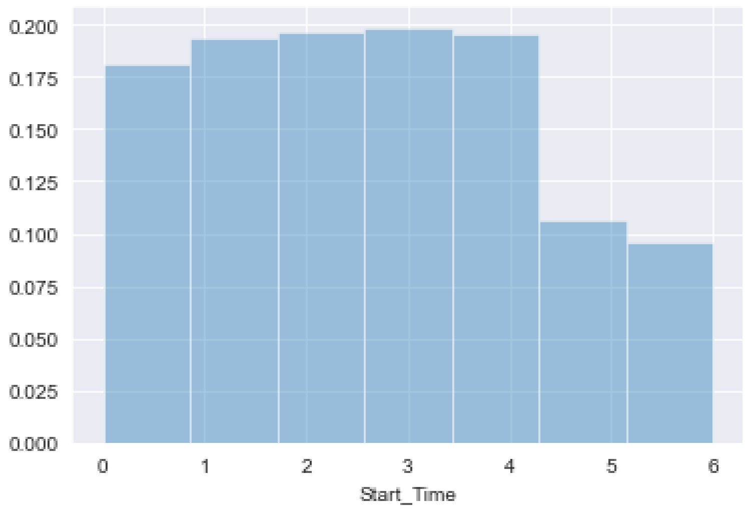
Day basis Accidents

In [20]:

```
sns.distplot(df.Start_Time.dt.dayofweek, bins=7, kde=False, norm_hist=True)
```

Out[20]:

```
<AxesSubplot:xlabel='Start_Time'>
```



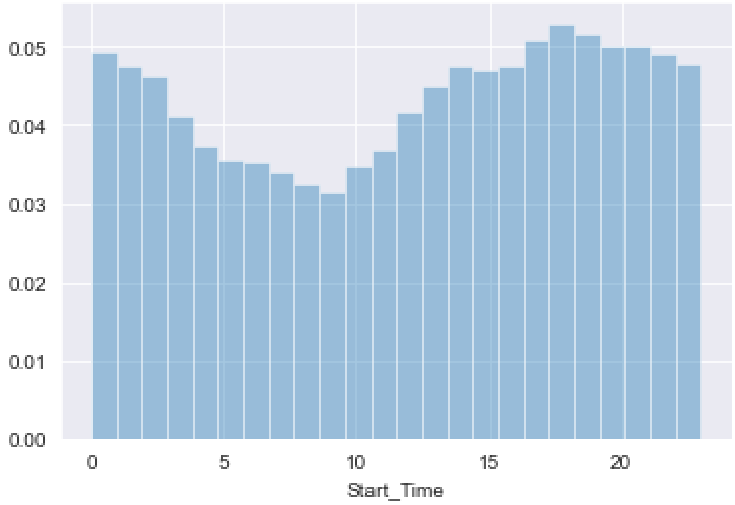
- Is the distrobution of accidents by hour the same on weekends as on weekdays

In [21]:

```
sunday_start_time = df.Start_Time[df.Start_Time.dt.dayofweek==6 ]
sns.distplot(sunday_start_time.dt.hour, bins=24, kde=False, norm_hist=True)
```

Out[21]:

```
<AxesSubplot:xlabel='Start_Time'>
```

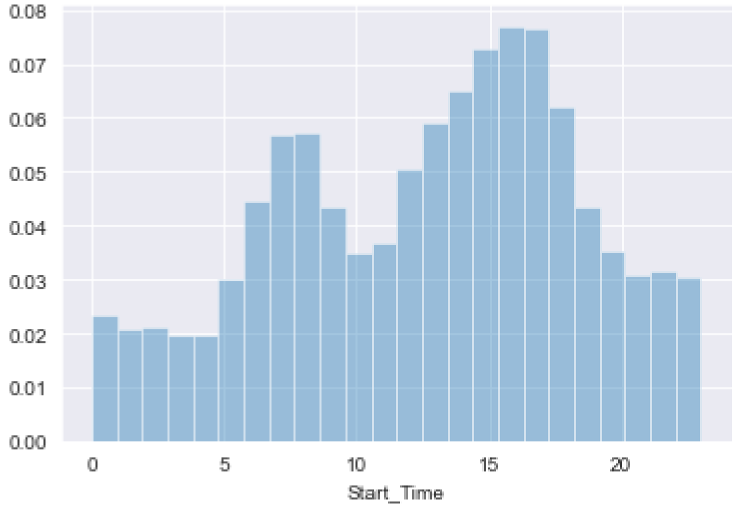


In [22]:

```
Monday_start_time = df.Start_Time[df.Start_Time.dt.dayofweek==0 ]
sns.distplot(Monday_start_time.dt.hour, bins=24, kde=False, norm_hist=True)
```

Out[22]:

```
<AxesSubplot:xlabel='Start_Time'>
```



- On sundays the peak of accidents occur between 10AM to 11PM, unlike Weekdays

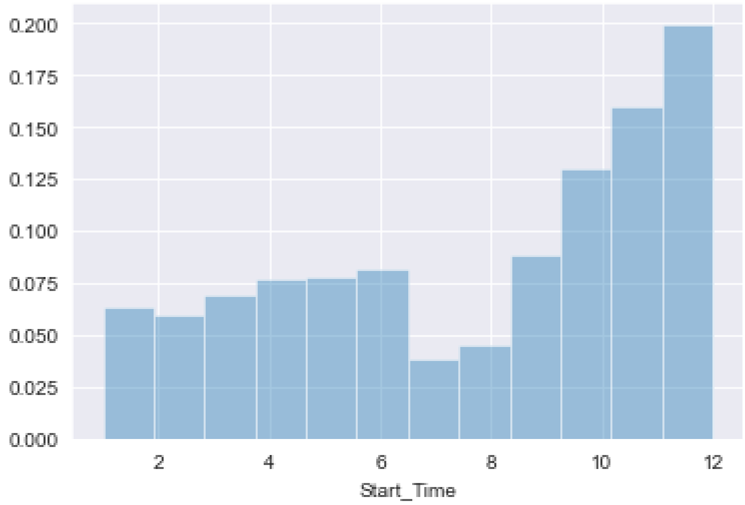
Monthly basis Accidents

In [23]:

```
sns.distplot(df.Start_Time.dt.month, bins=12, kde=False, norm_hist=True)
```

Out[23]:

```
<AxesSubplot:xlabel='Start_Time'>
```



- In general, most traffic fatalities occur in the summer and fall and that is the reason the vehicles goes slow. but in winter and in holidays the traffic is less than usual. And people drives faster and weekdays. That is the reason the most of the accidents occur between OCT to NOV but the in DEC it is the heighest

Start Latitude and Longitude

In [24]:

```
df.Start_Lat
```

Out[24]:

```
0          40.10891
1          39.86542
2          39.10266
3          39.10148
4          41.06213
             ...
1516059    34.00248
1516060    32.76696
1516061    33.77545
1516062    33.99246
1516063    34.13393
Name: Start_Lat, Length: 1516064, dtype: float64
```

In [25]:

```
df.Start_Lng
```

Out[25]:

```
0          -83.09286
1          -84.06280
2          -84.52468
3          -84.52341
4          -81.53784
             ...
1516059   -117.37936
1516060   -117.14806
1516061   -117.84779
1516062   -118.40302
1516063   -117.23092
Name: Start_Lng, Length: 1516064, dtype: float64
```

In [26]:

```
sample_df = df.sample(int(0.1 * len(df)))
sample_df
```

Out[26]:

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	I

	ID	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng
172969	A-2889576	3	2017-02-15 01:31:19	2017-02-15 07:31:19	33.910490	-112.145540	33.918960	-112.144380
1120715	A-3844056	2	2019-11-13 09:43:00	2019-11-13 11:15:44	38.885870	-121.339982	38.885870	-121.339982
919135	A-3642444	2	2020-04-15 12:06:37	2020-04-15 12:41:35	40.489640	-111.940000	40.489640	-111.940000
839855	A-3563135	2	2020-06-19 10:38:30	2020-06-19 11:08:30	37.635110	-77.459160	37.635110	-77.459160
832350	A-3555630	3	2020-06-16 08:32:02	2020-06-16 09:02:02	39.101660	-94.679310	39.101660	-94.679310
...	...	...	...	...	...	...	...	...
953190	A-3676517	4	2020-01-07 07:42:37	2020-01-07 08:11:09	47.979520	-122.185940	47.979120	-122.184340
1443176	A-4166519	2	2019-04-06 21:51:00	2019-04-07 01:51:00	44.915452	-122.988381	44.916948	-122.988850
191939	A-2908546	2	2017-05-03 17:23:33	2017-05-03 23:23:33	39.416140	-77.438440	39.416140	-77.438440
1297452	A-4020794	2	2018-07-09 08:25:35	2018-07-09 14:25:35	32.774840	-96.790160	32.772693	-96.798535
1205791	A-3929132	2	2019-01-18 22:09:35	2019-01-19 02:09:35	42.178597	-124.358030	42.174247	-124.357927

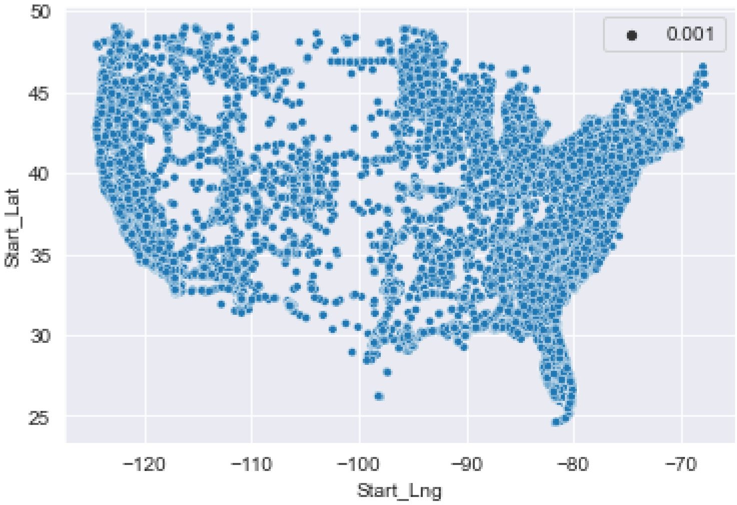
151606 rows × 47 columns

In [27]:

```
sns.scatterplot(x=sample_df.Start_Lng, y= sample_df.Start_Lat, size = 0.001)
```

Out[27]:

```
<AxesSubplot:xlabel='Start_Lng', ylabel='Start_Lat'>
```



In [28]:

```
!pip install folium
import folium
```

```
Requirement already satisfied: folium in c:\users\vitrogene\anaconda3\lib\site-packages
(0.12.1.post1)
Requirement already satisfied: requests in c:\users\vitrogene\anaconda3\lib\site-packages
(from folium) (2.26.0)
Requirement already satisfied: numpy in c:\users\vitrogene\anaconda3\lib\site-packages (fr
om folium) (1.20.3)
Requirement already satisfied: branca>=0.3.0 in c:\users\vitrogene\anaconda3\lib\site-pack
ages (from folium) (0.4.2)
Requirement already satisfied: jinja2>=2.9 in c:\users\vitrogene\anaconda3\lib\site-packag
es (from folium) (2.11.3)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\vitrogene\anaconda3\lib\site-p
ackages (from jinja2>=2.9->folium) (1.1.1)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\vitrogene\anaconda3\l
ib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\vitrogene\anaconda3\lib\s
ite-packages (from requests->folium) (1.26.7)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vitrogene\anaconda3\lib\site-packa
ges (from requests->folium) (3.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vitrogene\anaconda3\lib\site
-packages (from requests->folium) (2021.10.8)
```

In [29]:

```
from folium import plugins
from folium.plugins import HeatMap
```

In [30]:

```
sample_df = df.sample(int(0.001 * len(df)))
lat_lon_pairs = list(zip(list(df.Start_Lat), list(df.Start_Lng)))
```

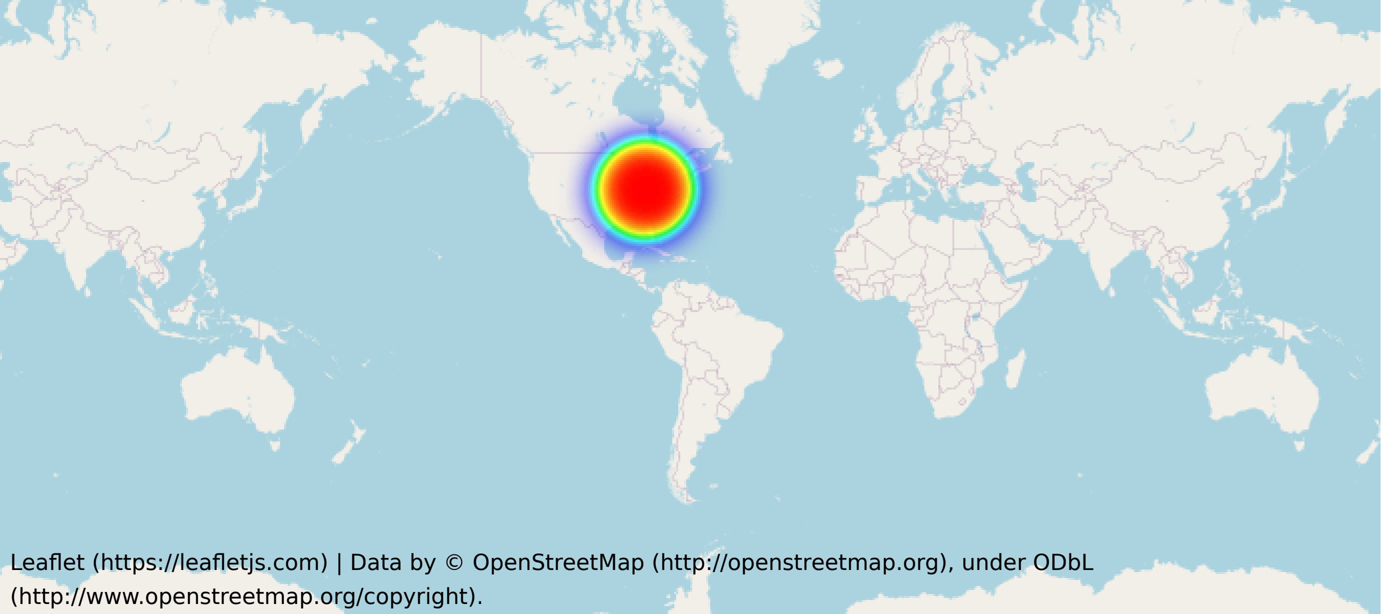
100 sample Accidents

In [31]:

```
map1= folium.Map()
HeatMap(lat_lon_pairs[:100]).add_to(map1)
map1
```

Out[31]: Make this Notebook Trusted to load map: File -> Trust Notebook

Loading [MathJax]/extensions/Safe.js


Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

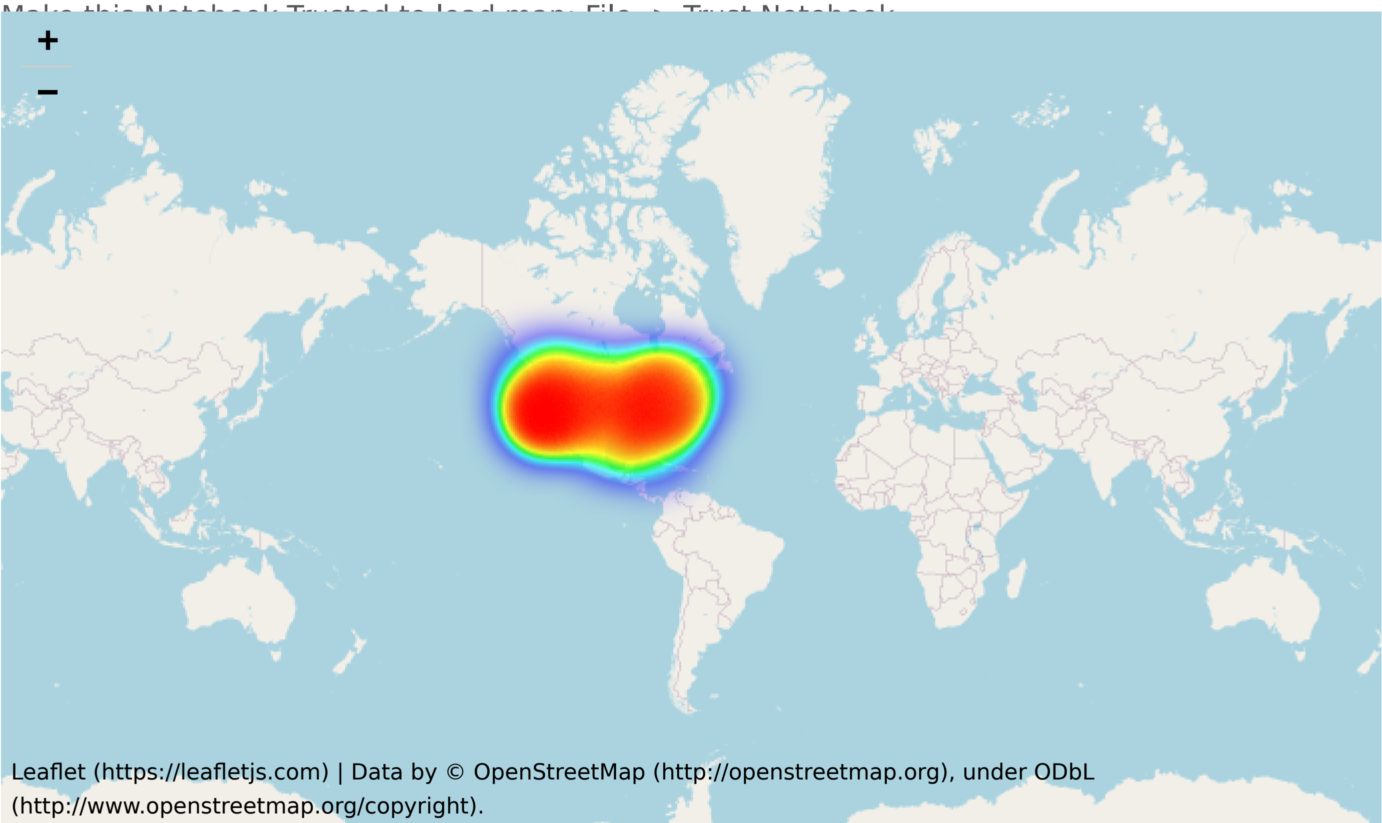
In [34]:

```
sample_df1 = df.sample(int(0.001 * len(df)))
lat_lon_pairs1 = list(zip(list(sample_df1.Start_Lat), list(sample_df1.Start_Lng)))
```

In [35]:

```
map2= folium.Map()
HeatMap(lat_lon_pairs1).add_to(map2)
map2
```

Out[35]: Make this Notebook Trusted to load map: File -> Trust Notebook


+
−
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [32]:

```
zip(list(df.Start_Lat), list(df.Start_Lng))
```

Out[32]:

```
<zip at 0x1a396042d40>
```

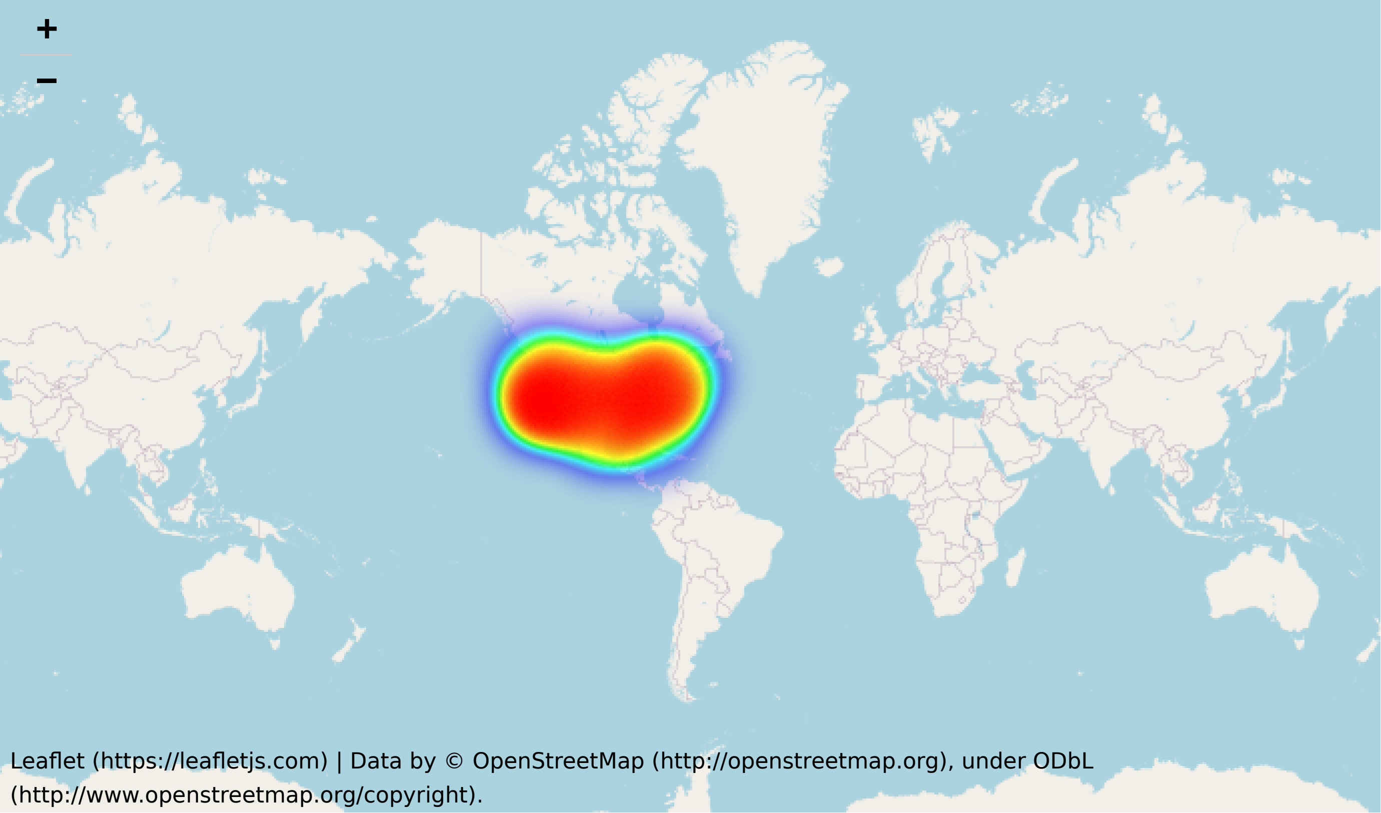
In [36]:

```
map= folium.Map()
HeatMap(zip(list(df.Start_Lat), list(df.Start_Lng))).add_to(map)
```

Loading [MathJax]/extensions/Safe.js

```
map
```

Out[36]: Make this Notebook Trusted to load map: File -> Trust Notebook


+
−
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [ ]:

Smmary and Conclusion

Insights:

- No Data for New York Althouth It is a big City.
- The number of accident per city decreases Exponentially.
- less than 3% of have more than 1000 yearly accidents.
- Over 1200 cities have reported just 1 accident
- deep orange areas are the most accident prone area

Loading [MathJax]/extensions/Safe.js