Analysis - US car accidents

March 16, 2020

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_style('whitegrid')
%matplotlib inline
```

[2]:
```
from urllib.request import urlopen
import json
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/
 ↪geojson-counties-fips.json') as response:
    counties = json.load(response)

import plotly.express as px
```

[3]:
```
import plotly.graph_objects as go
```

[4]:
```
pd.set_option('display.max_columns',None)
```

[5]:
```
ac = pd.read_csv('US_Accidents_Dec19.csv')
```

[6]:
```
ac.head()
```

[6]:

	ID	Source	TMC	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	Distance(mi)
0	A-1	MapQuest	201.0	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	NaN	NaN	0.01
1	A-2	MapQuest	201.0	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	NaN	NaN	0.01
2	A-3	MapQuest	201.0	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	NaN	NaN	0.01
3	A-4	MapQuest	201.0	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	NaN	NaN	0.01
4	A-5	MapQuest	201.0	2	2016-02-08 07:39:07	2016-02-08 08:09:07	39.627781	-84.188354	NaN	NaN	0.01

```
                                        Description  Number  \
0  Right lane blocked due to accident on I-70 Eas...     NaN
1  Accident on Brice Rd at Tussing Rd. Expect del...  2584.0
2  Accident on OH-32 State Route 32 Westbound at ...     NaN
3  Accident on I-75 Southbound at Exits 52 52B US...     NaN
4  Accident on McEwen Rd at OH-725 Miamisburg Cen...     NaN

                     Street Side          City      County State     Zipcode  \
0                     I-70 E    R        Dayton  Montgomery    OH       45424
1                  Brice Rd    L  Reynoldsburg    Franklin    OH  43068-3402
2             State Route 32    R  Williamsburg    Clermont    OH       45176
3                     I-75 S    R        Dayton  Montgomery    OH       45417
4  Miamisburg Centerville Rd    R        Dayton  Montgomery    OH       45459

  Country    Timezone Airport_Code    Weather_Timestamp  Temperature(F)  \
0      US  US/Eastern         KFFO  2016-02-08 05:58:00            36.9
1      US  US/Eastern         KCMH  2016-02-08 05:51:00            37.9
2      US  US/Eastern         KI69  2016-02-08 06:56:00            36.0
3      US  US/Eastern         KDAY  2016-02-08 07:38:00            35.1
4      US  US/Eastern         KMGY  2016-02-08 07:53:00            36.0

   Wind_Chill(F)  Humidity(%)  Pressure(in)  Visibility(mi) Wind_Direction  \
0            NaN         91.0         29.68            10.0           Calm
1            NaN        100.0         29.65            10.0           Calm
2           33.3        100.0         29.67            10.0             SW
3           31.0         96.0         29.64             9.0             SW
4           33.3         89.0         29.65             6.0             SW

   Wind_Speed(mph)  Precipitation(in) Weather_Condition  Amenity   Bump  \
0              NaN               0.02        Light Rain    False  False
1              NaN               0.00        Light Rain    False  False
2              3.5                NaN          Overcast    False  False
3              4.6                NaN     Mostly Cloudy    False  False
4              3.5                NaN     Mostly Cloudy    False  False

   Crossing  Give_Way  Junction  No_Exit  Railway  Roundabout  Station   Stop  \
0     False     False     False    False    False       False    False  False
1     False     False     False    False    False       False    False  False
2     False     False     False    False    False       False    False  False
3     False     False     False    False    False       False    False  False
4     False     False     False    False    False       False    False  False

   Traffic_Calming  Traffic_Signal  Turning_Loop Sunrise_Sunset  \
0            False           False         False          Night
1            False           False         False          Night
2            False            True         False          Night
```

```
3            False           False         False          Night
4            False            True         False            Day

  Civil_Twilight Nautical_Twilight Astronomical_Twilight
0          Night             Night                 Night
1          Night             Night                   Day
2          Night               Day                   Day
3            Day               Day                   Day
4            Day               Day                   Day
```

[7]:
```
ac.describe()
```

[7]:
```
                TMC      Severity     Start_Lat     Start_Lng        End_Lat  \
count  2.246264e+06  2.974335e+06  2.974335e+06  2.974335e+06  728071.000000
mean   2.078316e+02  2.360190e+00  3.649361e+01 -9.542625e+01      37.580871
std    2.032959e+01  5.414733e-01  4.918849e+00  1.721881e+01       5.004757
min    2.000000e+02  1.000000e+00  2.455527e+01 -1.246238e+02      24.570110
25%    2.010000e+02  2.000000e+00  3.355040e+01 -1.172920e+02      33.957554
50%    2.010000e+02  2.000000e+00  3.584969e+01 -9.025083e+01      37.903670
75%    2.010000e+02  3.000000e+00  4.037026e+01 -8.091891e+01      41.372630
max    4.060000e+02  4.000000e+00  4.900220e+01 -6.711317e+01      49.075000

             End_Lng  Distance(mi)        Number  Temperature(F)  \
count  728071.000000  2.974335e+06  1.056730e+06    2.918272e+06
mean      -99.976032  2.855654e-01  5.837004e+03    6.235120e+01
std        18.416647  1.548392e+00  1.515928e+04    1.878855e+01
min      -124.497829  0.000000e+00  0.000000e+00   -7.780000e+01
25%      -118.286610  0.000000e+00  8.370000e+02    5.000000e+01
50%       -96.631690  0.000000e+00  2.717000e+03    6.440000e+01
75%       -82.323850  1.000000e-02  7.000000e+03    7.600000e+01
max       -67.109242  3.336300e+02  9.999997e+06    1.706000e+02

       Wind_Chill(F)   Humidity(%)  Pressure(in)  Visibility(mi)  \
count   1.121712e+06  2.915162e+06  2.926193e+06    2.908644e+06
mean    5.132685e+01  6.540542e+01  2.983190e+01    9.150770e+00
std     2.519127e+01  2.255676e+01  7.213808e-01    2.892114e+00
min    -6.590000e+01  1.000000e+00  0.000000e+00    0.000000e+00
25%     3.200000e+01  4.900000e+01  2.982000e+01    1.000000e+01
50%     5.400000e+01  6.700000e+01  2.998000e+01    1.000000e+01
75%     7.300000e+01  8.400000e+01  3.011000e+01    1.000000e+01
max     1.150000e+02  1.000000e+02  3.304000e+01    1.400000e+02

       Wind_Speed(mph)  Precipitation(in)
count     2.533495e+06      975977.000000
mean      8.298064e+00           0.020495
std       5.138546e+00           0.235770
min       0.000000e+00           0.000000
```

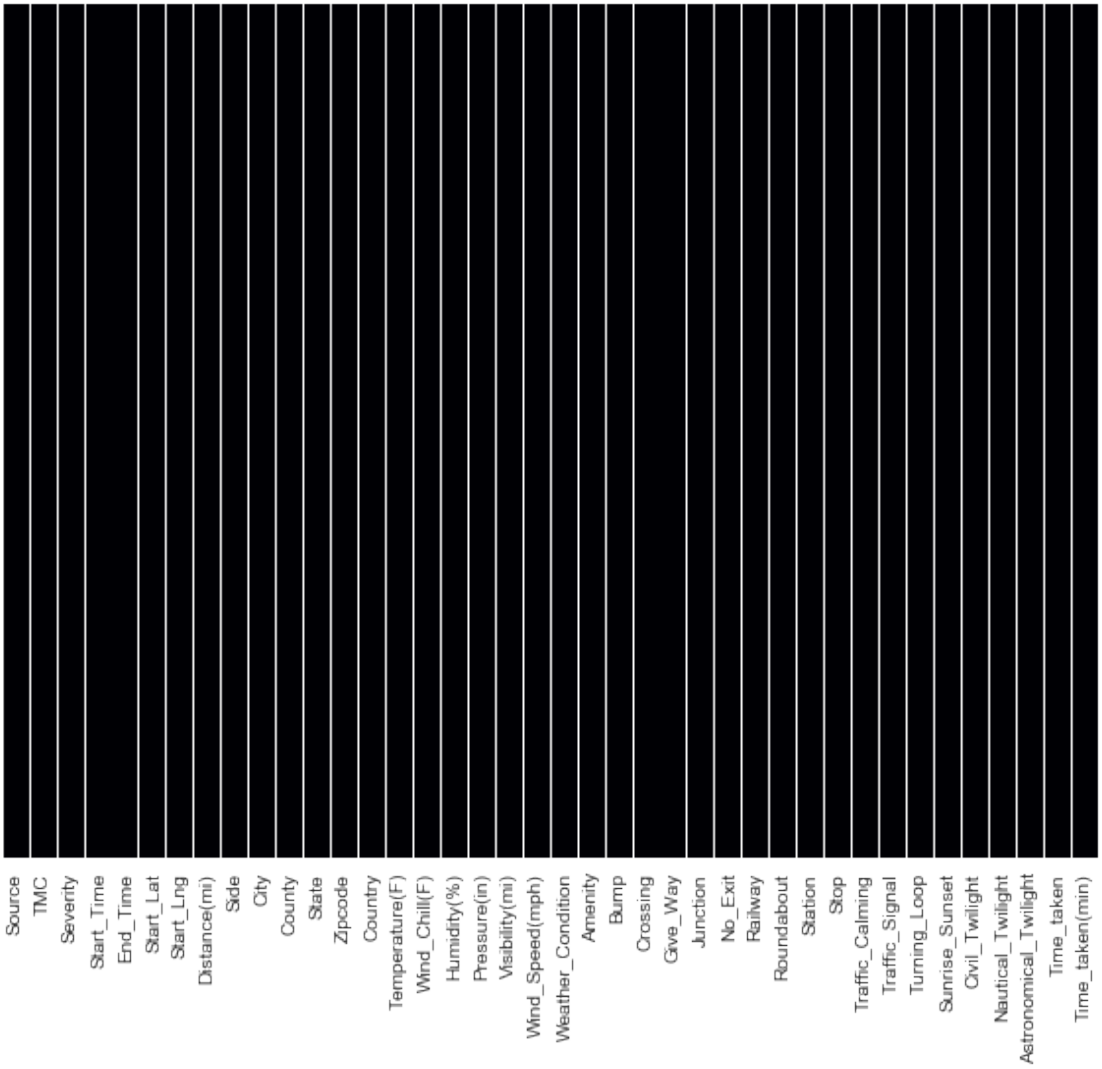
```
25%        4.600000e+00           0.000000
50%        7.000000e+00           0.000000
75%        1.040000e+01           0.000000
max        8.228000e+02          25.000000
```

[8]:
```
#preprocessing data

#drop unnecessary columns
at=ac.
 ↪drop(['End_Lat','End_Lng','Description','Number','Street','Airport_Code','ID','Timezone','Wea

#fill na values with corresponding mean value.
at['Wind_Chill(F)'].fillna(51.32685,inplace=True)
at['Temperature(F)'].fillna(62.35120,inplace=True)
at['Humidity(%)'].fillna(65.40542,inplace=True)
at['Pressure(in)'].fillna(29.83190,inplace=True)
at['Visibility(mi)'].fillna(9.150770,inplace=True)
at['Wind_Speed(mph)'].fillna(8.298064,inplace=True)
at['TMC'].fillna(207.8316,inplace=True)
at['Zipcode'].fillna(0,inplace=True)

#grouping weather conditions
at['Weather_Condition'].fillna('Clear',inplace=True)
at.loc[at['Weather_Condition'].str.contains('snow', case=False),␣
 ↪'Weather_Condition'] = 'Snow'
at.loc[at['Weather_Condition'].str.contains('Rain', case=False),␣
 ↪'Weather_Condition'] = 'Rain'
at.loc[at['Weather_Condition'].str.contains('Drizzle', case=False),␣
 ↪'Weather_Condition'] = 'Drizzle'
at.loc[at['Weather_Condition'].str.contains('Cloudy', case=False),␣
 ↪'Weather_Condition'] = 'Cloudy'
at.loc[at['Weather_Condition'].str.contains('Fair', case=False),␣
 ↪'Weather_Condition'] = 'Fair'
at.loc[at['Weather_Condition'].str.contains('Fog', case=False),␣
 ↪'Weather_Condition'] = 'Fog'
at.loc[at['Weather_Condition'].str.contains('Ice Pellets', case=False),␣
 ↪'Weather_Condition'] = 'Ice Pellets'
at.loc[at['Weather_Condition'].str.contains('Sleet', case=False),␣
 ↪'Weather_Condition'] = 'Sleet'
at.loc[at['Weather_Condition'].str.contains('Smoke', case=False),␣
 ↪'Weather_Condition'] = 'Smoke'
at.loc[at['Weather_Condition'].str.contains('T-Storm', case=False),␣
 ↪'Weather_Condition'] = 'Thunderstorms '
at.loc[at['Weather_Condition'].str.contains('Thunderstorms ', case=False),␣
 ↪'Weather_Condition'] = 'Thunderstorms '
```

```
at.loc[at['Weather_Condition'].str.contains('Sand', case=False),␣
 ↪'Weather_Condition'] = 'Sand'
at.loc[at['Weather_Condition'].str.contains('Cloud', case=False),␣
 ↪'Weather_Condition'] = 'Cloudy'
at.loc[at['Weather_Condition'].str.contains('Windy', case=False),␣
 ↪'Weather_Condition'] = 'Windy'
at.loc[at['Weather_Condition'].str.contains('Squalls', case=False),␣
 ↪'Weather_Condition'] = 'Squalls'
at.loc[at['Weather_Condition'].str.contains('Thunder', case=False),␣
 ↪'Weather_Condition'] = 'Thunder'
at.loc[at['Weather_Condition'].str.contains('Dust', case=False),␣
 ↪'Weather_Condition'] = 'Dust'
at.loc[at['Weather_Condition'].str.contains('N/A Precipitation', case=False),␣
 ↪'Weather_Condition'] = 'Clear'

#creating a column of time taken to clean up accident scene
at['Start_Time'] = pd.to_datetime(at['Start_Time'])
at['End_Time'] = pd.to_datetime(at['End_Time'])
at['Time_taken'] = (at['Start_Time'] - at['End_Time']).abs().dt.seconds
at.loc[(at['Time_taken'] > 0), 'Time_taken(min)'] = (at['Time_taken'] / 60)
at['Time_taken(min)'] = at['Time_taken(min)'].round()
```

[9]:
```
#graphing new columns NA values in each categories.
plt.figure(figsize=(10,8))
sns.heatmap(at.isna(),yticklabels=False,cbar=False,cmap='magma')
```

[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x24c1cd0c0c8>
```



[10]:
```
at.head()
```

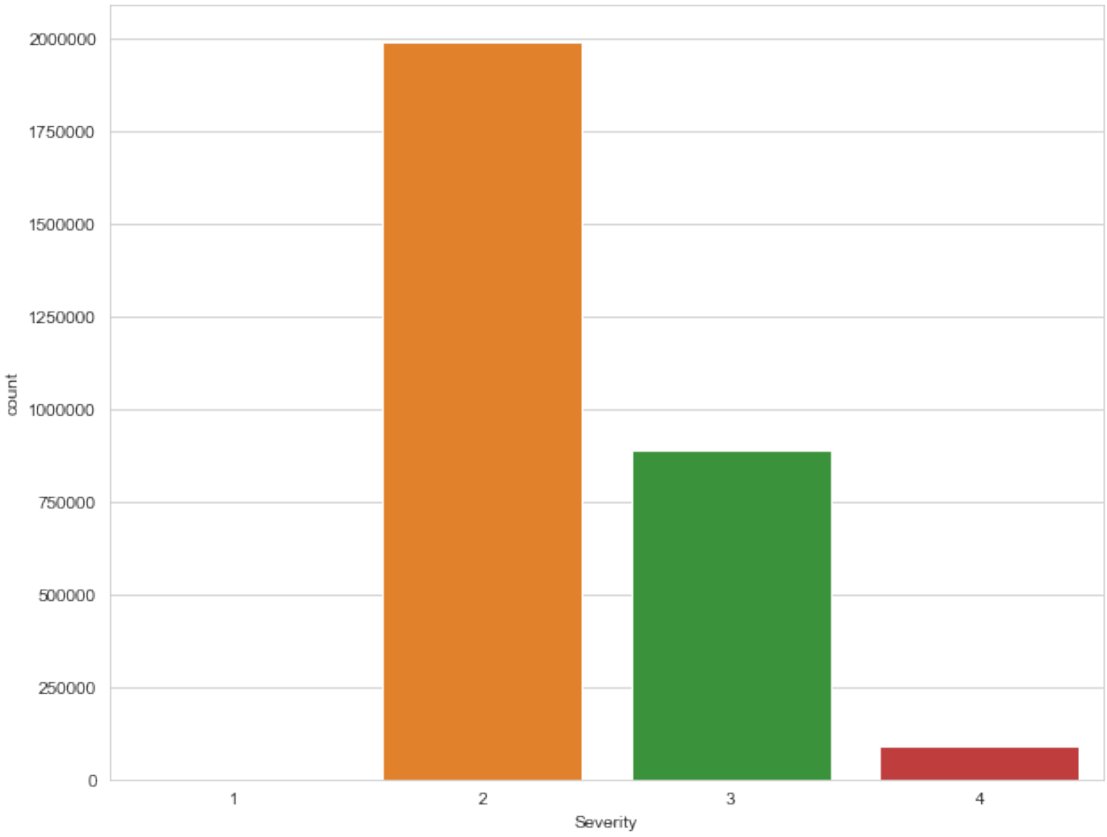
[10]:

	Source	TMC	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	Distance(mi)	Side	City	County	State
0	MapQuest	201.0	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	0.01	R	Dayton	Montgomery	OH
1	MapQuest	201.0	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	0.01	L	Reynoldsburg	Franklin	OH
2	MapQuest	201.0	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	0.01	R	Williamsburg	Clermont	OH
3	MapQuest	201.0	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	0.01	R	Dayton	Montgomery	OH
4	MapQuest	201.0	2	2016-02-08 07:39:07	2016-02-08 08:09:07							

```
4  39.627781 -84.188354          0.01    R        Dayton  Montgomery    OH

      Zipcode Country  Temperature(F)  Wind_Chill(F)  Humidity(%)  \
0       45424      US            36.9       51.32685         91.0
1  43068-3402      US            37.9       51.32685        100.0
2       45176      US            36.0       33.30000        100.0
3       45417      US            35.1       31.00000         96.0
4       45459      US            36.0       33.30000         89.0

   Pressure(in)  Visibility(mi)  Wind_Speed(mph) Weather_Condition  Amenity  \
0         29.68            10.0         8.298064              Rain    False
1         29.65            10.0         8.298064              Rain    False
2         29.67            10.0         3.500000          Overcast    False
3         29.64             9.0         4.600000            Cloudy    False
4         29.65             6.0         3.500000            Cloudy    False

    Bump  Crossing  Give_Way  Junction  No_Exit  Railway  Roundabout  Station  \
0  False     False     False     False    False    False       False    False
1  False     False     False     False    False    False       False    False
2  False     False     False     False    False    False       False    False
3  False     False     False     False    False    False       False    False
4  False     False     False     False    False    False       False    False

    Stop  Traffic_Calming  Traffic_Signal  Turning_Loop Sunrise_Sunset  \
0  False            False           False         False          Night
1  False            False           False         False          Night
2  False            False            True         False          Night
3  False            False           False         False          Night
4  False            False            True         False            Day

  Civil_Twilight Nautical_Twilight Astronomical_Twilight  Time_taken  \
0          Night             Night                 Night       18840
1          Night             Night                   Day        1800
2          Night               Day                   Day        1800
3            Day               Day                   Day        1800
4            Day               Day                   Day        1800

   Time_taken(min)
0            314.0
1             30.0
2             30.0
3             30.0
4             30.0
```

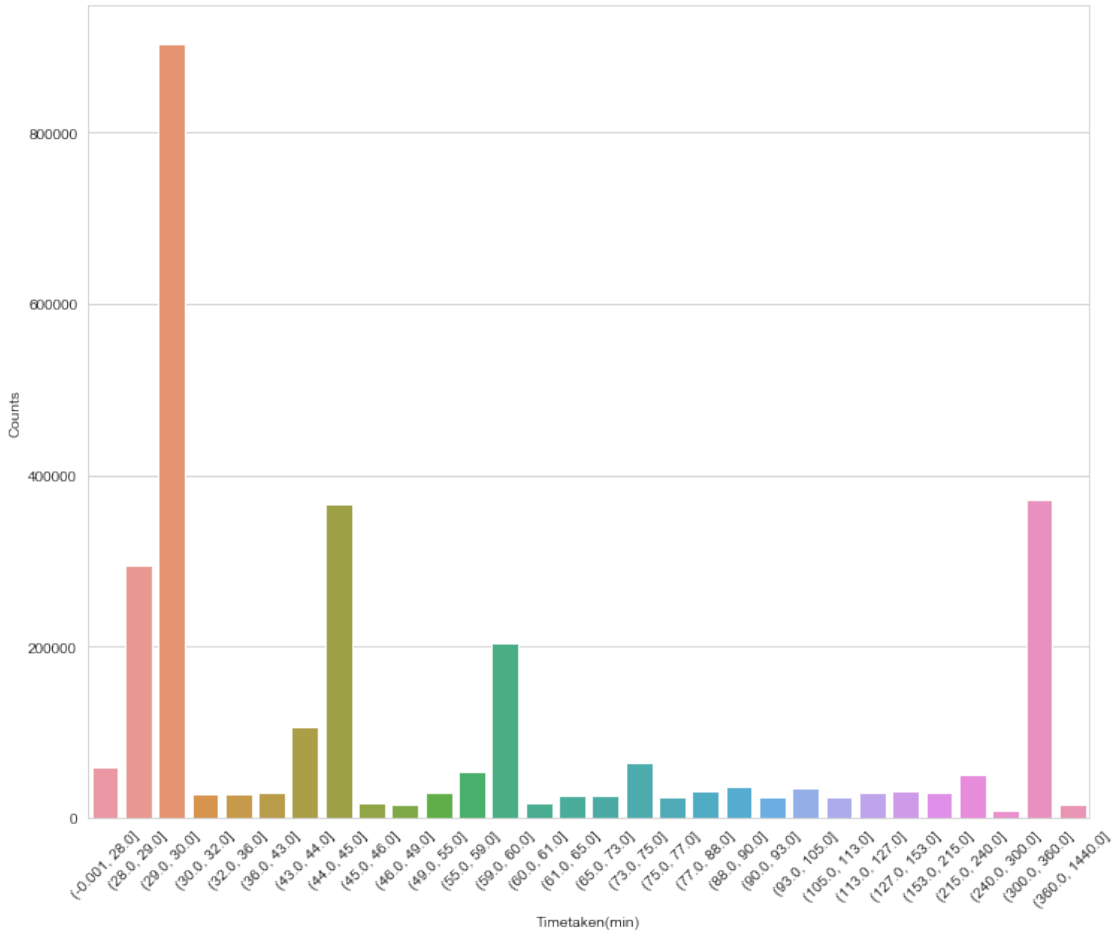
[11]:
```
#counts of accident severity level
plt.figure(figsize=(10,8))
sns.countplot(at['Severity'])
```

[11]: <matplotlib.axes._subplots.AxesSubplot at 0x24c25a23e88>



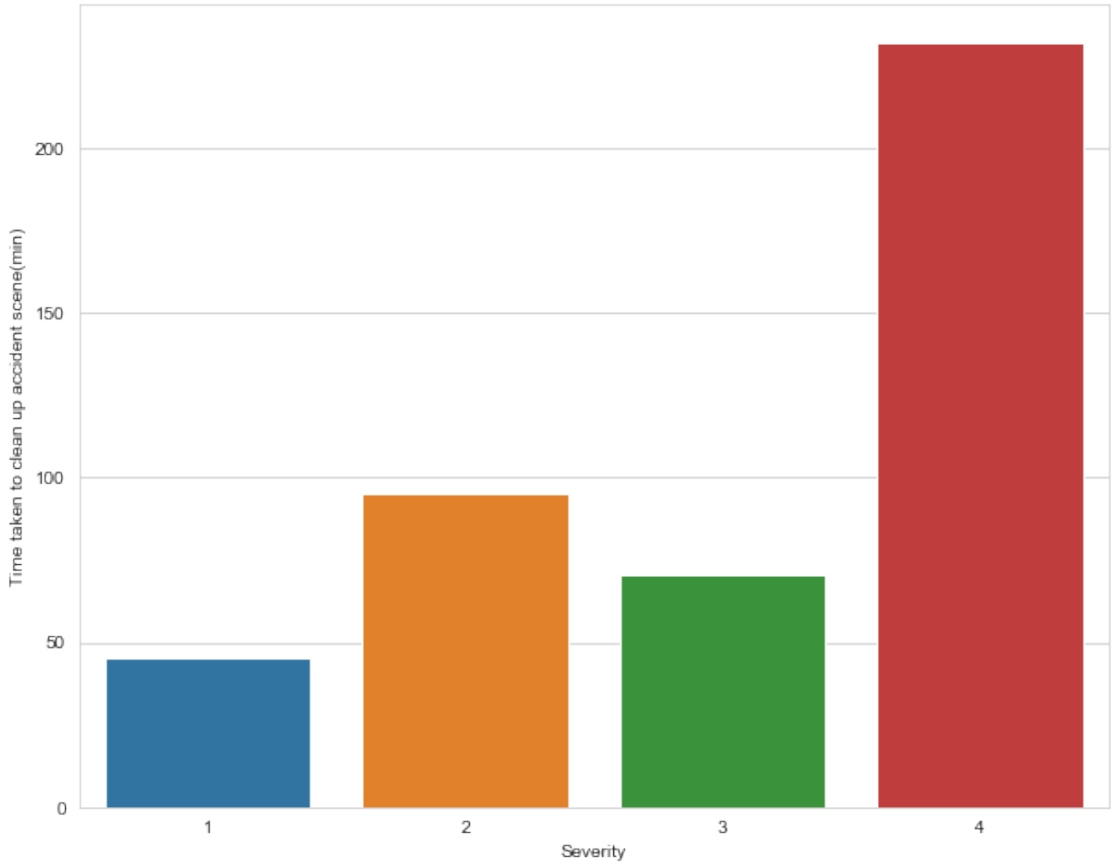
[152]:
```
#groupby accident scene time taken(min) and count of it
tttmean = at.groupby(pd.qcut(at['Time_taken(min)'],q=100,duplicates='drop')).
 ↪count()
tttmean.index.name = 'Timetaken(min)'
tttmean=tttmean.reset_index()
plt.figure(figsize=(12,10))
sns.barplot(x=tttmean['Timetaken(min)'],y=tttmean['Source'])
plt.xticks(rotation=45)
plt.ylabel('Counts')
```

[152]: Text(0, 0.5, 'Counts')

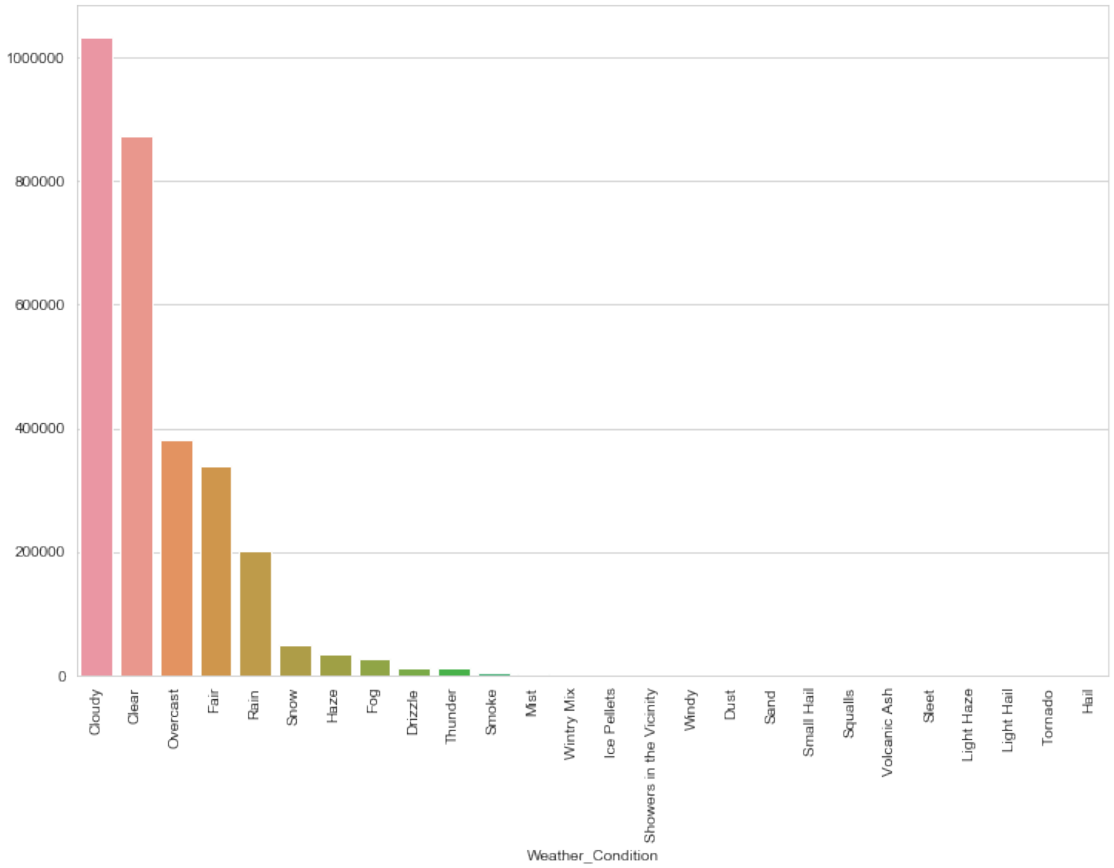


[13]:
```
#Time taken by severity level.
sevmean = at.groupby(at['Severity'])['Time_taken(min)'].mean()
plt.figure(figsize=(10,8))
sns.barplot(x=sevmean.index,y=sevmean.values)
plt.ylabel('Time taken to clean up accident scene(min)')
```

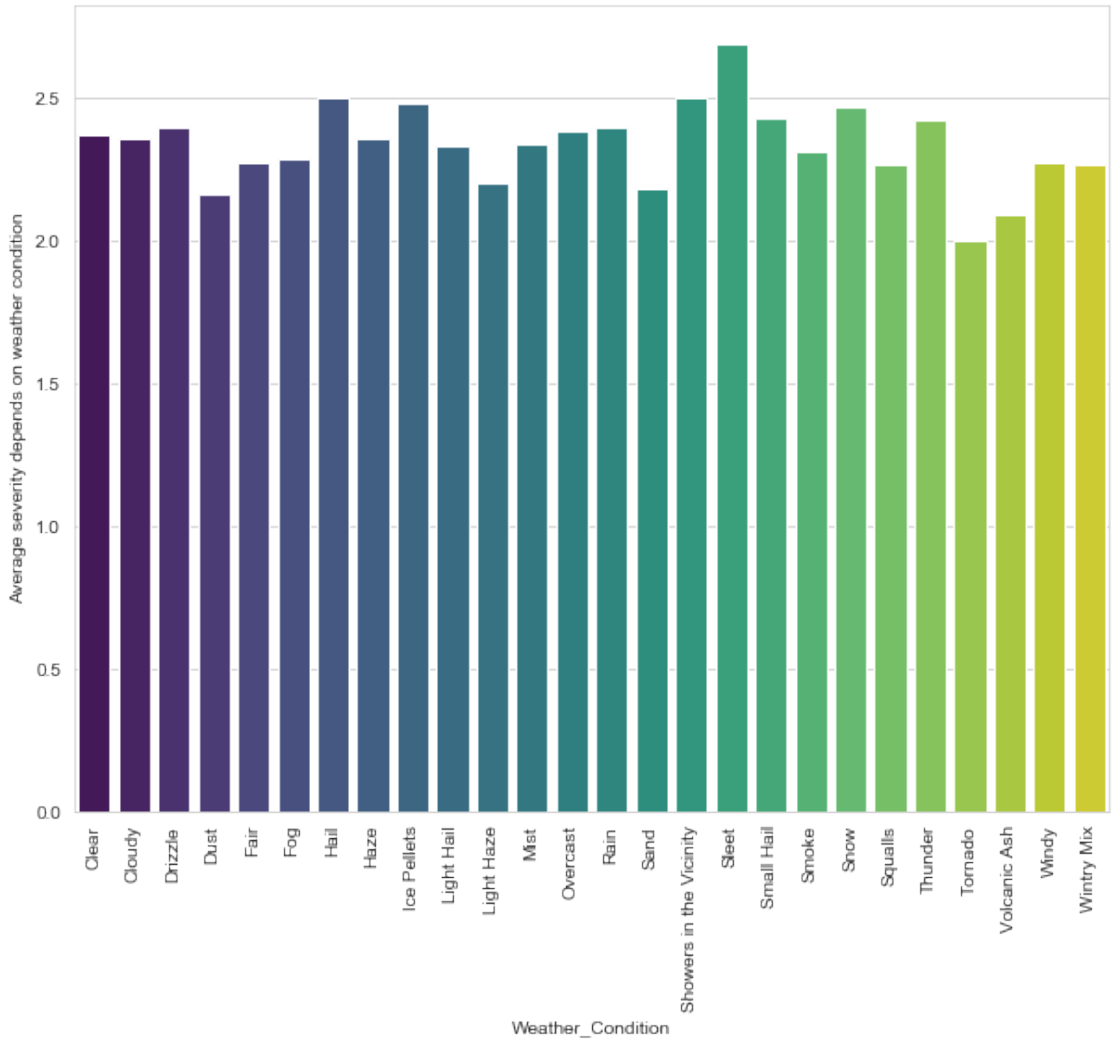
[13]:
```
Text(0, 0.5, 'Time taken to clean up accident scene(min)')
```



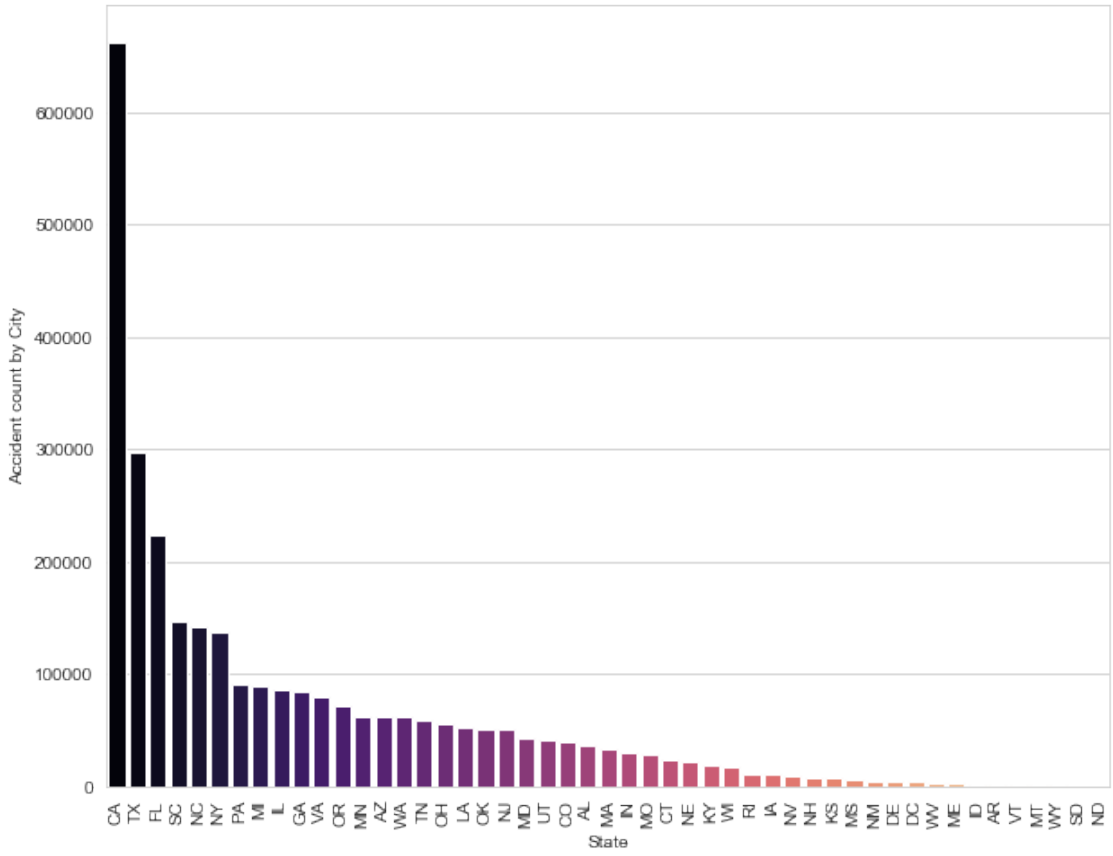
[14]:
```
#accidents count by weather condition.
wtct = at.groupby(at['Weather_Condition']).count()
wtct=wtct.sort_values('Source', ascending=False)
plt.figure(figsize=(12,8))
sns.barplot(x=wtct['Source'].index,y=wtct['Source'].values)
plt.xticks(rotation=90)
plt.show()
```



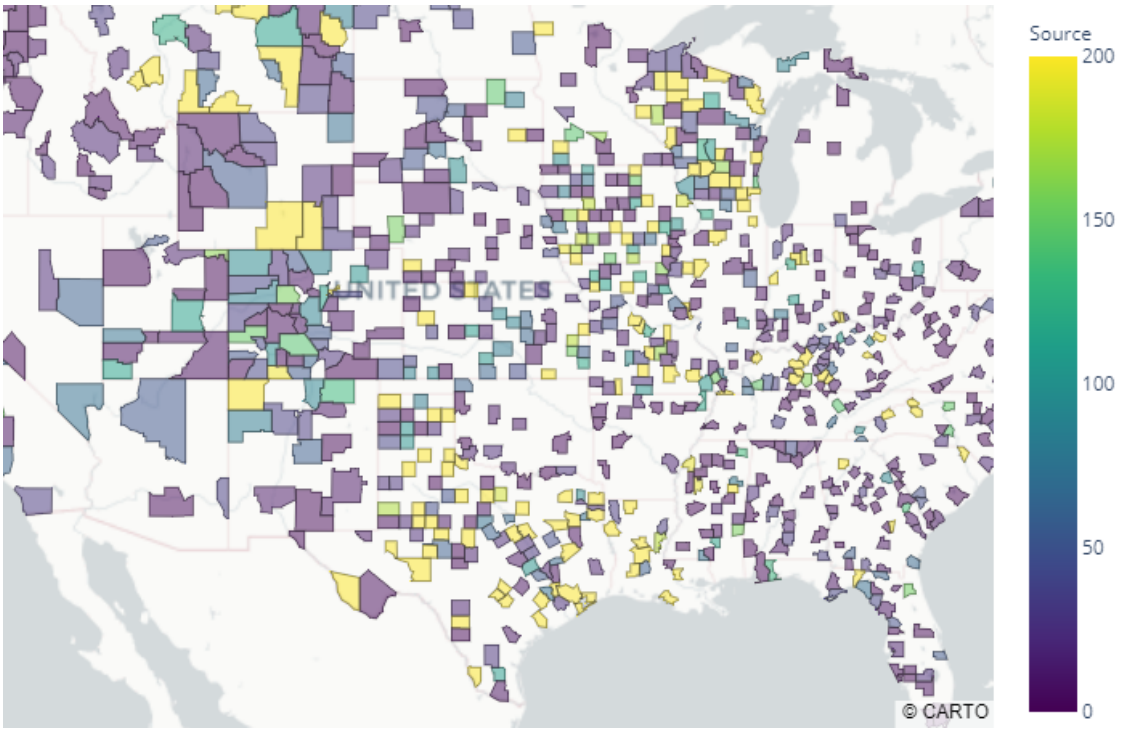
[15]:
```
#Average severity depends on weather condition
wtmean = at.groupby(at['Weather_Condition'])['Severity'].mean()
plt.figure(figsize=(10,8))
sns.barplot(x=wtmean.index,y=wtmean.values,palette='viridis')
plt.ylabel('Average severity depends on weather condition')
plt.xticks(rotation=90)
plt.show()
```



[16]:
```
#Accident counts by States
ctct = at.groupby(at['State']).count()
ctct=ctct.sort_values('Source', ascending=False)
plt.figure(figsize=(10,8))
sns.barplot(x=ctct['Source'].index,y=ctct['Source'].values,palette='magma')
plt.ylabel('Accident count by City')
plt.xticks(rotation=90)
plt.show()
```



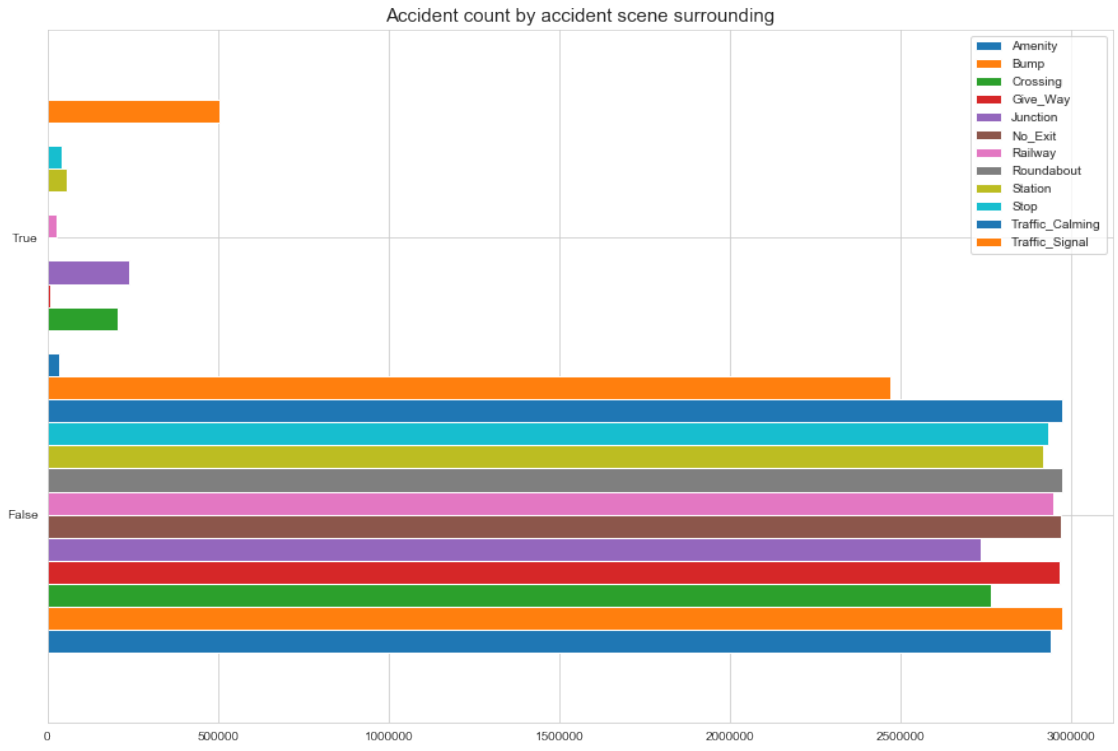
[17]:
```
#Map accident counts by zipcode
zt = at.groupby(at['Zipcode']).count().reset_index()
fig = px.choropleth_mapbox(zt, geojson=counties, locations=zt['Zipcode'],␣
 ↪color=zt['Source'],
                           color_continuous_scale="Viridis",
                           range_color=(0, 200),
                           mapbox_style="carto-positron",
                           zoom=3, center = {"lat": 37.0902, "lon": -95.7129},
                           opacity=0.5,
                           labels={'accident count by zipcode'}
                          )
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



[18]:
```
#count true/false on each boolean columns
a1=at['Amenity'].value_counts().reset_index()
a2=at['Bump'].value_counts().reset_index()
a3=at['Crossing'].value_counts().reset_index()
a4=at['Give_Way'].value_counts().reset_index()
a5=at['Junction'].value_counts().reset_index()
a6=at['No_Exit'].value_counts().reset_index()
a7=at['Railway'].value_counts().reset_index()
a8=at['Roundabout'].value_counts().reset_index()
a9=at['Station'].value_counts().reset_index()
a10=at['Stop'].value_counts().reset_index()
a11=at['Traffic_Calming'].value_counts().reset_index()
a12=at['Traffic_Signal'].value_counts().reset_index()
a13=at['Turning_Loop'].value_counts().reset_index()
tfdata=pd.merge(a1,a2)
tfdata=pd.merge(tfdata,a3)
tfdata=pd.merge(tfdata,a4)
tfdata=pd.merge(tfdata,a5)
tfdata=pd.merge(tfdata,a6)
tfdata=pd.merge(tfdata,a7)
tfdata=pd.merge(tfdata,a8)
tfdata=pd.merge(tfdata,a9)
tfdata=pd.merge(tfdata,a10)
tfdata=pd.merge(tfdata,a11)
```

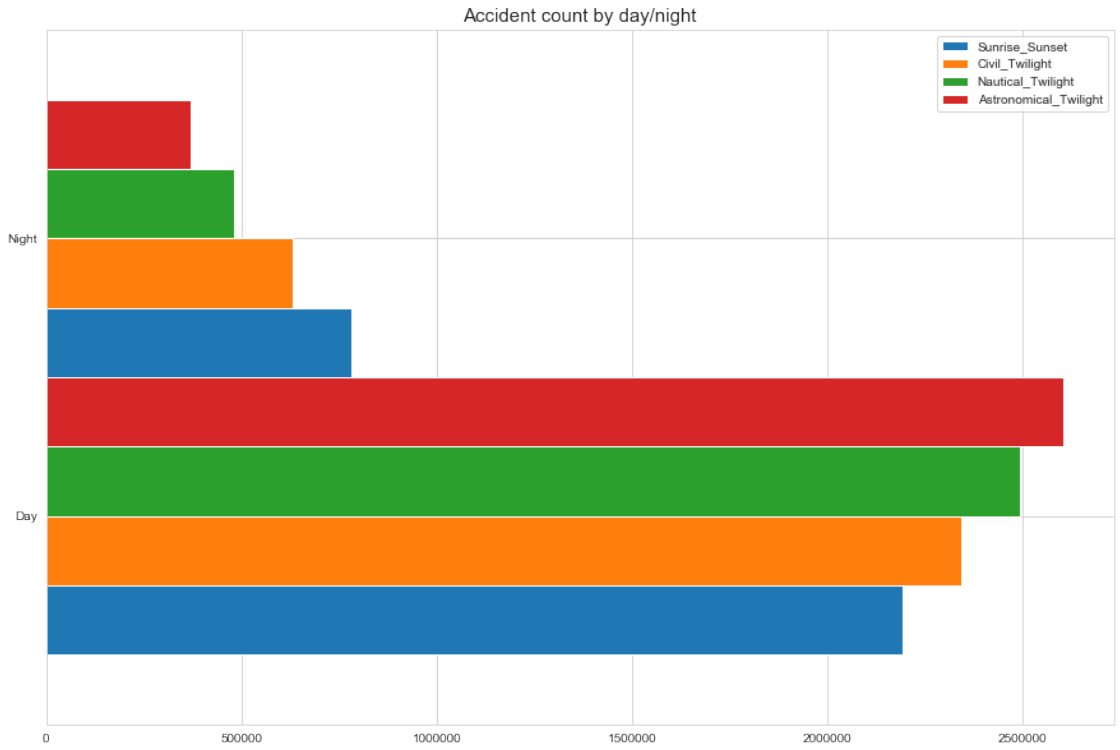
```
tfdata=pd.merge(tfdata,a12)
```

[19]:
```
#plot true false count on each category.
tfdata.plot.barh(figsize=(15,10),width=1)
plt.yticks([1,0],['True','False'])
plt.title('Accident count by accident scene surrounding',fontsize=15)
plt.show()
```

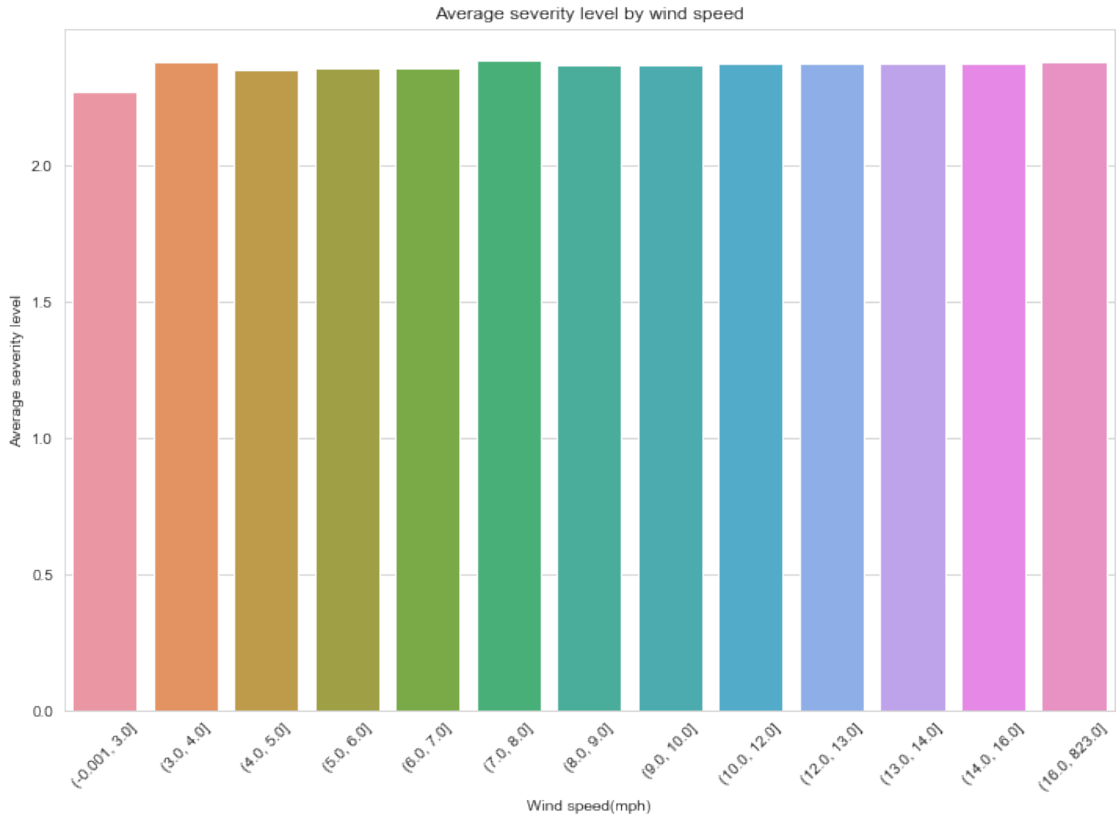


[20]:
```
#count day/night on each columns
b1=at['Sunrise_Sunset'].value_counts().reset_index()
b2=at['Civil_Twilight'].value_counts().reset_index()
b3=at['Nautical_Twilight'].value_counts().reset_index()
b4=at['Astronomical_Twilight'].value_counts().reset_index()
tfdata2=pd.merge(b1,b2)
tfdata2=pd.merge(tfdata2,b3)
tfdata2=pd.merge(tfdata2,b4)
```

[21]:
```
#plot day night count on each category.
tfdata2.plot.barh(figsize=(15,10),width=1)
plt.yticks([1,0],['Night','Day'])
plt.title('Accident count by day/night',fontsize=15)
plt.show()
```



[153]:
```
#plot average severity level depends on wind speed
at['Wind_Speed(mph)'] =at['Wind_Speed(mph)'].round()
wtt=at.groupby(pd.
 ↪qcut(at['Wind_Speed(mph)'],q=20,duplicates='drop'))['Severity'].mean().
 ↪reset_index()
plt.figure(figsize=(12,8))
sns.barplot(x=wtt['Wind_Speed(mph)'],y=wtt['Severity'])
plt.title('Average severity level by wind speed')
plt.ylabel('Average severity level')
plt.xlabel('Wind speed(mph)')
plt.xticks(rotation=45)
plt.show()
```


Average severity level by wind speed
Average severity level
2.0
1.5
1.0
0.5
0.0
(-0.001, 3.0]
(3.0, 4.0]
(4.0, 5.0]
(5.0, 6.0]
(6.0, 7.0]
(7.0, 8.0]
(8.0, 9.0]
(9.0, 10.0]
(10.0, 12.0]
(12.0, 13.0]
(13.0, 14.0]
(14.0, 16.0]
(16.0, 823.0]
Wind speed(mph)

[ ]: