Assessment-2: Data Informed Essay

Predicting Gun Violence in the U.S. – A Decomposable Time Series Approach

Lukas Birkenmaier, Student ID: u2085320

Centre for Interdisciplinary Methodologies, University of Warwick

IM939 – Data Science across Disciplines: Principles, Practice and Critique

Associate Professor Cagatay Turkay

12th January 2020

Word Count: 1990 (2000 max.)

Abstract

On January 6, 2021, armed protesters invaded the U.S. Capitol, which ultimately lead to violent shootings and several people being shot or injured. However, this event is only one among numerous incidents of gun-related violence in United States every day. In this data informed essay, I aim to shed light on the temporal patterns of gun-related violence and aim to explain varying levels of gun-violence over time. Doing so, I use a decomposable time series model and analyze the time-related components of gun-violence simultaneously. Moreover, I then use this model to make predictions beyond the data available. My findings show that gun-related incidents in the U.S. follow a strong periodical temporal pattern and my model forecast lower levels of gun-violence in the future, which may be shaped by non-time related explanatory factors like the implementation of gun control laws.

Table of Contents

Assessment 2

Predicting Gun-Related Violence in the United States

Introduction

Gun related violence is a serious public health problem in the United States (Cook & Ludwig, 2000; Gabor, 2016). According to research from Grinshteyn & Hemenway (2016), overall U.S. gun homicide rate was found to be 25 times higher compared to other high-income countries, with an even higher rate of 49 in the category of 15- to 24-year-olds. Evaluating those incidents in terms of reoccurring patterns, data scientists can extract valuable information for law enforcement agencies. In particular, temporal data from previous gun-related incidents could be used to predict future cases, thereby enabling law enforcement agencies to effectively deploy resources and improve security (Wang & Brown, 2012). Starting from the question how time and gun-violence in the U.S are related, I will apply a decomposable time series model to examine temporal patterns within the recorded incidents (cf. Harvey & Peters, 1990). Doing so, I will not only describe recent trends in U.S. gun-related incidents, but will use this model to make predictions about future incidents as well.

Following this brief introduction, section two gives a short overview over the phenomenon of U.S. gun violence and its time related dimension. Afterwards, section three introduces the data as well as the forecasting model (cf. Taylor & Letham, 2018). Section four then presents the main results of the analysis. At the end, section five discusses those results and their limitations and identifies possible next steps in specifying the model even further.

Understanding U.S. Gun Violence

In the past, several scholars have tried to explain varying levels of gun violence in the United States. According to researchers, possible factors to affect the likelihood of gun incidents might include (male) gender (Page, 2009), the amount of social networks usage (Bond & Bushman, 2017), individual fascination with firearms (Meloy et al., 2004), the overall availability of guns (Hemenway & Miller, 2000; Lee et al., 2017) and the number of firearm laws (Fleegler et al., 2013) as well as " socioeconomic disparity, instability, inequality, lack of democratic processes, health, social and educational policies" (Cukier & Eagen, 2018, p. 111). Building up on this research, scholars have recently started to incorporate temporal explanations for crime-related incidents as well. Fowler et al. (2015), for instance, evaluate

temporal trends in firearm death and injury rates between 1993 and 2012 and find overall stable rates within the time period under study. Evaluating the temporal trend of gun violence in the medium-sized town Syracuse, New York, Larsen et al. (2017) find evidence that the level of gun violence "remained relatively stable with more gunshots in the warmer months […] and fewer gunshots in the colder months" (p. 4). As the most challenging model identified in my literature review to explain gun-incidents, Wang & Brown (2012) develop a spatio-temporal generalized additive model (ST-GAM) to predict gun-incidents. Evaluating the performance of their model, they find that reoccurring temporal and spatial patterns are a strong predictor for different levels of gun violence.

Research Design

To answer my research question on how gun violence can be explained by temporal factors, I rely on data from the non-profit organization Gun Violence Archive, which, according to their website, provides the "best, most detailed, accessible data on the subject [of] gun violence [in the US]" (GVA - Gun Violence Archive, 2020). However, in my analysis, I rely on a modified dataset by James Ko (2018), who used web scraping techniques to condense all information from the GVA website into one single dataset. The dataset consists of 239.677-gun violence incidents in the U.S. between 2013 and 2018 and contains detailed information on the incident, including e. g. exact date, the number of people affected, or which type of gun was used. However, since the data from 2013 are incomplete, I drop all cases before January 1, 2014. As prediction horizon, I choose a 2-year period as the default option.

Since I expect my data to be influenced by multiple seasonality, trend changes and outliers, I apply a decomposable time series model embedded in the open-source library *Facebook Prophet* (Taylor & Letham, 2018). Similar to generalized additive models (GAM) (cf Hastie & Tibshirani, 1990), *Prophet* enables me to include several time-related components to model my dependent variable 'Number of gun-incidents'. The main components of the *Prophet* model are displayed in Equation 1, where g(t) represents the overall trend, s(t) represents periodic changes (weekly, monthly or yearly seasonality), h(t) represents a vector of holidays and $\in_t$ an error term.

$$y(t) = g(t) + s(t) + h(t) + \in_t \quad (1)$$

The linear and non-linear functions of time (components g(t), s(t) and h(t)) are then applied as "smoothers" to the (single) regressor time for the dependent variable y(t). By decomposing time related influence on gun violence into its different components, I am able to accommodate different model specifications and compare the influence of the components next to each other. In order to calculate the overall trend g(t), the model provides both a "saturating growth model [with a maximum carrying capacity C]" or a "piecewise linear model" (Robson, 2019). In their paper, Taylor & Letham (2018) provide concise and detailed information on the mathematical specification of all components.

In order to adjust the basic model to my research question, I first define the daily number of gun-related incidents in the U.S. as my primary dependent variable. To calculate my model, I rely on the piecewise linear model, which is not limited by a maximum carrying capacity, since there are, in theory, no upper limits for the amount of gun incidents. Furthermore, I include weekly and yearly seasonality as periodic components, since "seasonal effects naturally arise and can be expected in time series generated by human actions" (Taylor & Letham, 2018, p. 4). Next, I also include a list of repeating holiday events, such as New Year's, which are likely to affect varying levels of gun-violence across the country (for a list of all holidays, see Appendix 1). To test the performance of my model, I use the build in diagnostics of the *Prophet* library (cf. Appendix 2), which provides functionality for time series cross validation by evaluating the differences between the predicted and true values using common performance measurements (Taylor & Letham, 2020). In this essay, I will mainly rely on the mean absolute percentage error (MAPE) to measure model performance, which shows the accuracy of the prediction as a percentage of the overall error.

The analysis was conducted using a Jupyter Notebook in Python 3. A full copy of the code used in this analysis is displayed in Appendix 6.

Results

Data Exploration

Figure 1 displays the total number of incidents, the total number of people killed, and the total number of people injured per day in a single plot. Looking at the time period under study, one can see strong seasonality within the data. Especially the levels of gun-related incidents appear to follow a yearly pattern, with growing levels in the summer months and lower levels in spring and fall. Exploring the issue of seasonality more closely, Appendix 3 gives a complete overview over the weekly, monthly and yearly incidents for each unit.

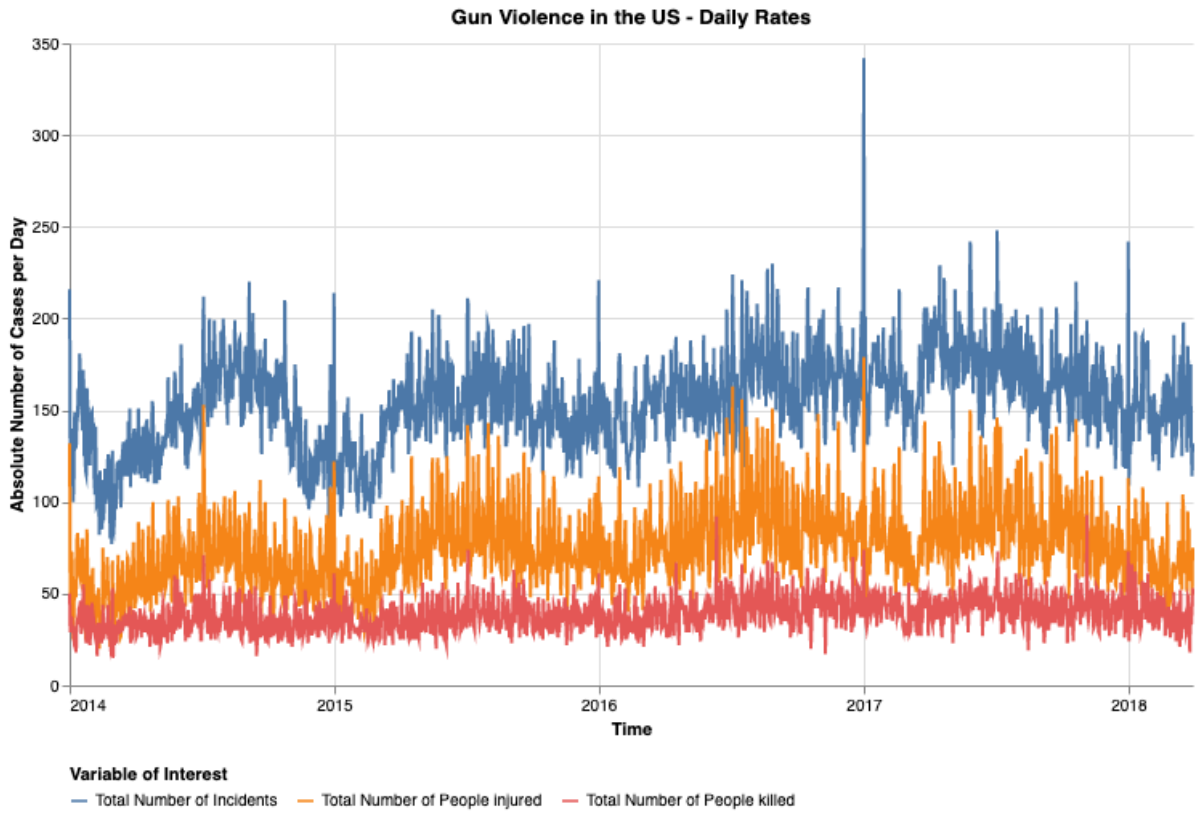


Figure 1: Absolute Daily Number of Gun-Incidents, Total Number of People Shot and Total Number of People Injured

Furthermore, it becomes visible that specific holidays have a big impact on the daily level of gun-related incidents within the year. For instance, looking at the most striking outliers in the time period, one can see a heavy spike at the turn of each year, which may be caused by excessive New Year celebrations. Inspecting the phenomenon of daily outliers more closely, Appendix 4 displays the 15 days with the highest and lowest levels of gun-related incidents. In summary, this explorative data inspection provides evidence that the numbers of gun-related incidents indeed follow a repeating temporal pattern.

Predicting Gun Incidents

Figure 3 displays the fitted model for the training and the prediction period. Black dots represent the recorded amounts of gun-incident, with most cases ranging between 100 to 200 gun-related incidents per day. Furthermore, the deep blue line represents the prediction of the model, the light blue shades visualize 95% confidence intervals around the predicted values and the red line represents the overall trend, where changes in the trend are highlighted by the vertical red dashed lines. The figure shows an overall positive trend of gun-related incidents until the end of 2017, with a significant changepoint and decreasing numbers in the time period afterwards. This negative trend consequently shapes the predicted values of the model as well, hence forecasting an overall decline of cases continuing until spring 2020.

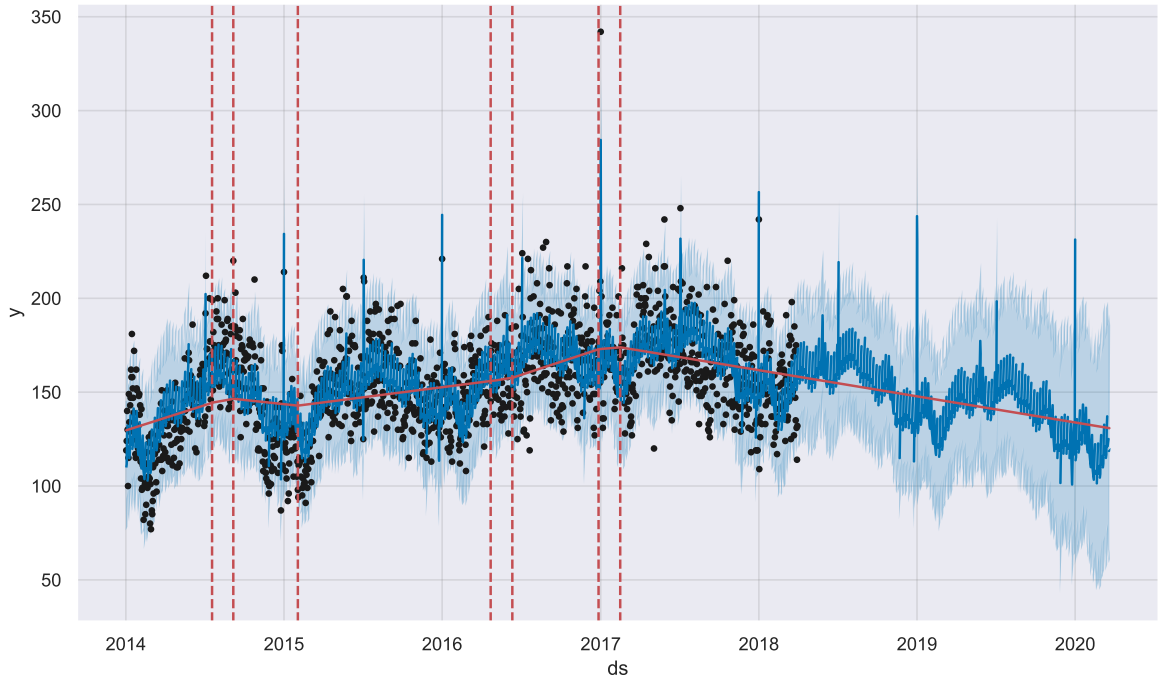


Figure 2: Fitted Model to the data

Evaluating the effect of temporal seasonality, Figure 4 displays the subplots for all components of the model. The first subplot again shows the trend, whereas the second subplot visualizes the unique impact of holidays. In particular, one can see that cases spike on New Year's Eve each year, with *ceteris paribus* more than 100 incidents per day. However, the effect of holidays can also lead to a reduction of cases, since e. g. Christmas appears to have a negative impact of 20 cases per day. The third subplot shows the weekly seasonality, with higher levels of gun-violence on Saturdays and Sundays and lower levels throughout the week. The last subplot represents the yearly seasonality, showing that levels of gun-related incidents are

lowest at the end of February (approximately 20 fewer cases per day) and highest in the mid of July (with a peak of approximately 15 more incidents per day).

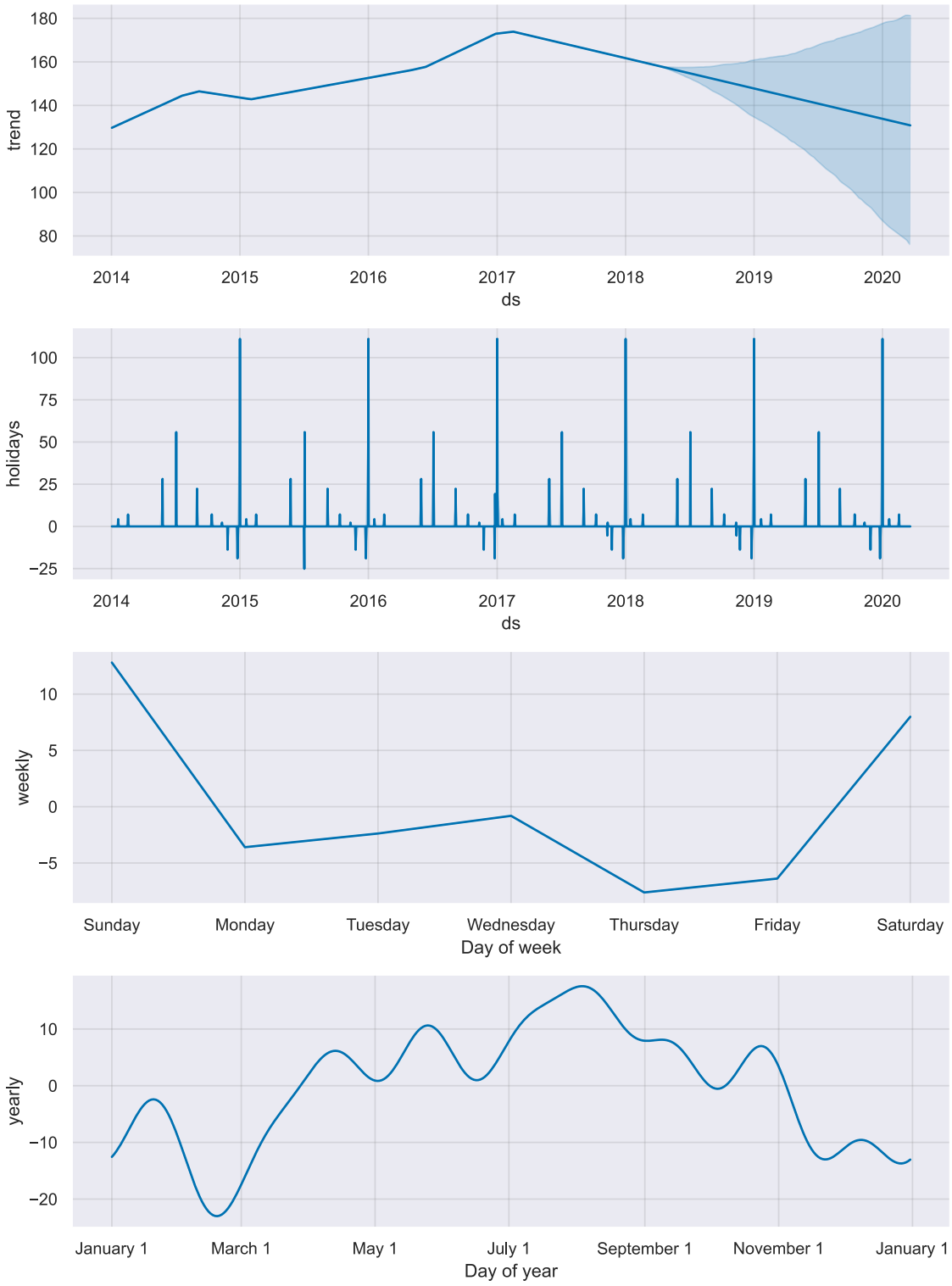


Figure 3: Single Components of the Model

Evaluating the performance of my model, Table 4 plots the mean absolute percentage error (MAPE) against the overall temporal horizon. Results show that my predicted values have an error rate between 10 and 20 percent, with better accuracy for shorter time periods (until 50 days: < 10%) and worse accuracy for longer time periods (after 150 days: >15%). These findings imply that the model performs well in forecasting levels of gun-incidents, with only minor errors being in an acceptable range. However, single model robustness metrics like MAPE should always be interpreted with utmost caution. This is, because the performance of a model is often not absolute and robustness metrics unfold their explanatory power best when comparing different specifications of one model next to each other (Khair et al., 2017). Appendix 5 gives a complete overview over alternative model performance metrices for the model under study.

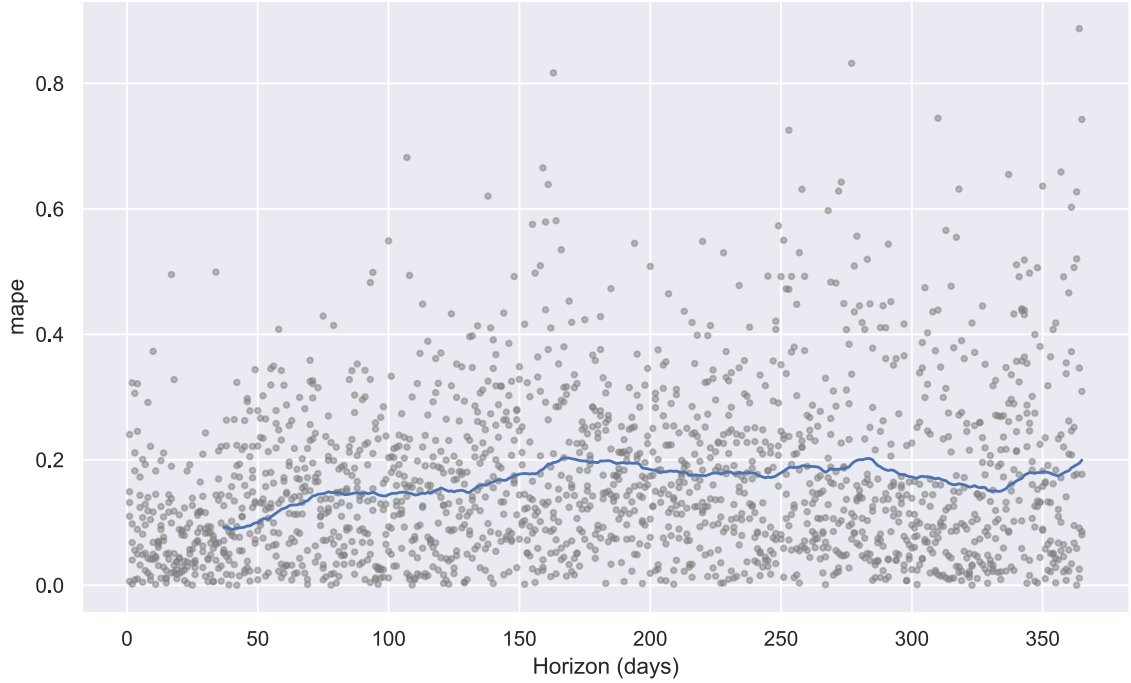


Figure 4: Plotting MAPE

Conclusion

Motivated by alarmingly high levels of gun-violence in the US, the goal of this essay has been to shed light on analyzing temporal patterns in the number of gun-related incidents. Results of my decomposable time series model show that cases indeed follow distinct weekly-, yearly- and holiday-seasonality, and the overall trend of gun-violence was predicted to decline for the forecasted period until 2020. However, these results can only be considered a starting point for further research. Without doubt, researchers should continue to examine the determinants

of gun violence more closely. For instance, the recent decline in gun-violence could be linked to increasing state-level initiatives to strengthen gun control laws, since "[in 2017,] more than 30 laws preventing gun violence passed in state legislatures" (Delkic, 2017).

Moreover, scholars should also take into account the severity of the incidents recorded, possibly by including the number of deaths per shooting as regression weights. Another possible step to improve model performance would be to include additional regressors, such as temperature or humidity. More generally, researchers should use this model to explain gun-violence by focusing on more local levels, thereby effectively controlling for regional and spatial characteristics.

Coming to an end, I want to reassure that we should not take gun-violence for granted. Because, as research has often shown (Hemenway & Miller, 2000; Lee et al., 2017), gun-related violence always arises from human behavior and, thus, can be reduced by implementing adequate policies, like for instance gun control laws or the prohibition of firearms.

Appendix 1

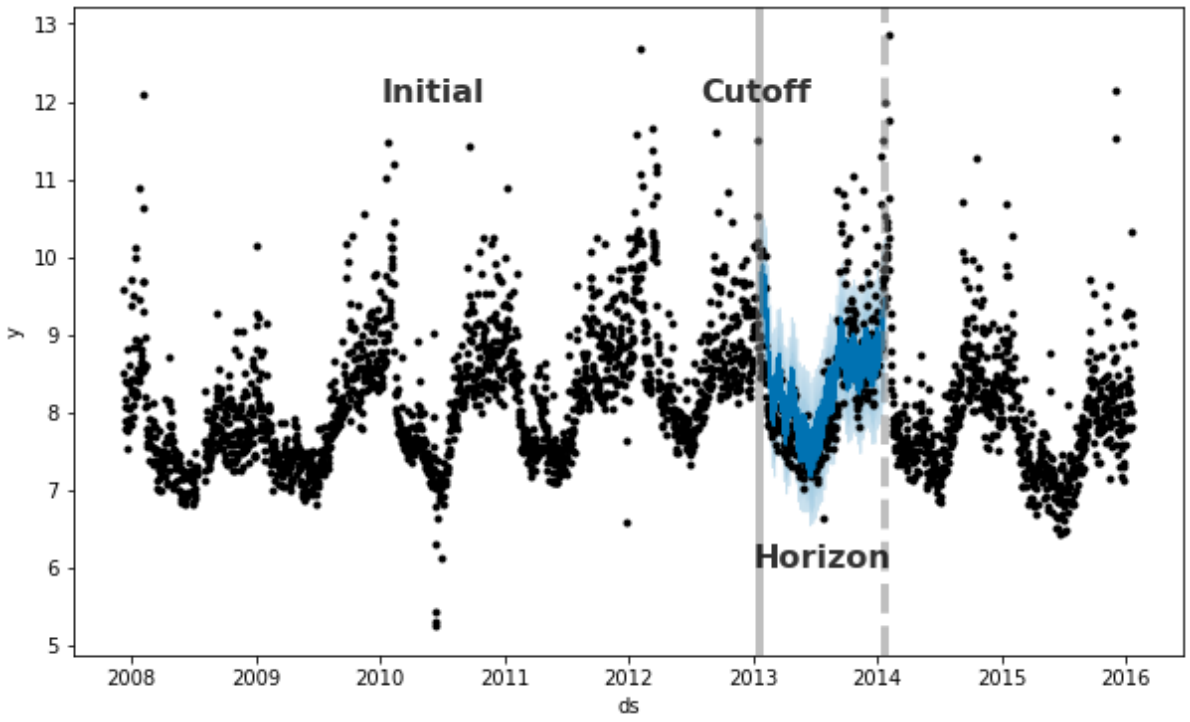
Title: Overview Holidays

0	New Year's Day
1	Martin Luther King Jr. Day
2	Washington's Birthday
3	Memorial Day
4	Independence Day
5	Labor Day
6	Columbus Day
7	Veterans Day
8	Thanksgiving
9	Christmas Day
10	Christmas Day (Observed)
11	New Year's Day (Observed)
12	Veterans Day (Observed)
13	Independence Day (Observed)

Note: The holidays for each country are provided by the holidays package in Python. A list of available countries, and the country name to use, is available on their page: https://github.com/dr-prodigy/python-holidays

Appendix 2

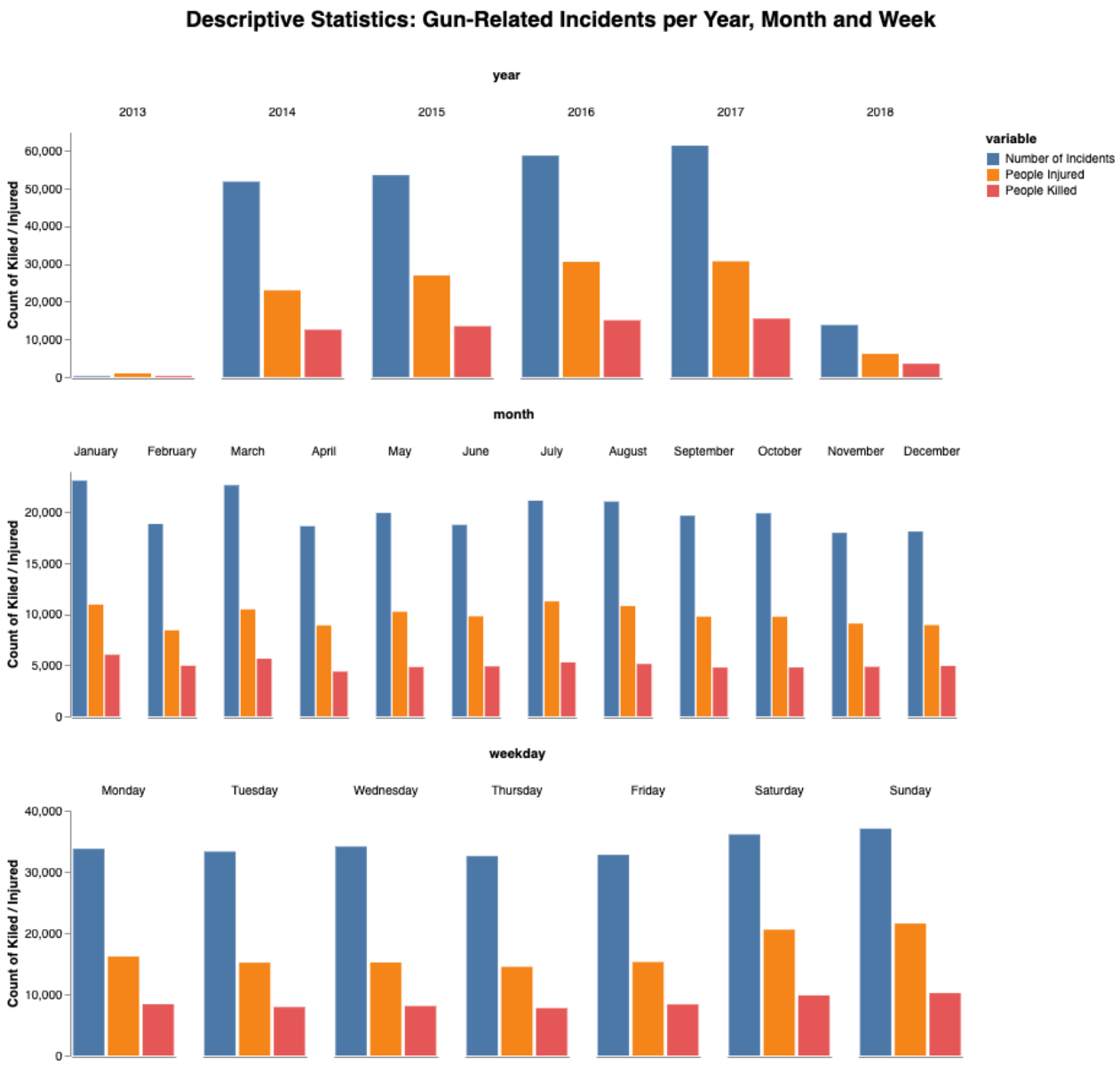
Title: Visual Display of Time Series Cross Validation



Note: Source: Taylor & Letham (2020) Retrieved January 10, 2021 from: https://face-book.github.io/prophet/docs/diagnostics.html

Appendix 3

Title: Displaying Number of People injured and killed across weeks, months and years



Appendix 4

Title: Extracting the days with the 20 highest and lowest number of cases

A. Days with the 20 highest cases of gun-related violence in the US

Nr	Date	# Incidents
1	2017-01-01	342
2	2017-07-04	248
3	2018-01-01	242
4	2017-05-28	242
5	2016-08-28	230
6	2017-04-16	229
7	2016-08-21	227
8	2016-07-04	224
9	2017-04-22	222
10	2016-01-01	221
11	2016-07-17	221
12	2014-09-06	220
13	2017-10-21	220
14	2016-11-27	217
15	2017-05-29	217

B. Days with the 20 lowest cases of gun-related violence in the US

Nr	Date	# Incidents
1	2014-02-28	77
2	2014-02-26	80
3	2014-02-11	82
4	2014-02-15	85
5	2014-03-03	85
6	2014-12-25	87
7	2014-03-02	87
8	2014-02-17	90
9	2015-02-20	91
10	2014-03-04	92
11	2015-01-10	92
12	2015-02-03	94
13	2015-02-12	95
14	2014-11-27	96
15	2015-01-12	96

Appendix 5

Title: Performance Metrics

Horizon	**MSE**	**RMSE**	**MAE**	**MAPE**	**MDAPE**	**COVER-AGE**
37 days	390.06	19.74	15.15	0.09	0.06	0.89
60 days	664.60	25.78	20.52	0.12	0.09	0.76
90 days	812.72	28.50	23.73	0.14	0.13	0.74
180 days	1425.52	37.75	31.45	0.19	0.15	0.56
360 days	1141.88	33.79	27.54	0.18	0.15	0.65

Note: MSE = Mean Squared Error, RMSE = Root Mean Square Error, MAE = Mean Absolute Error, MAPE = Mean Absolute Percentage Error, MDAPE = Median Absolute Percentage Error

Appendix 6

Title: Syntax

As requested, the complete Syntax of my analysis is attached at the end of this essay.

References

Bond, R. M. & Bushman, B. J. (2017). The contagious spread of violence among US adolescents through social networks. *American Journal of Public Health*, 107(2), 288–294.

Cook, P. J. & Ludwig, J. (2000). Gun violence: The real costs. Oxford University Press on Demand.

Cukier, W. & Eagen, S. A. (2018). Gun violence. *Current Opinion in Psychology*, 19, 109–112.

De Myttenaere, A., Golden, B., Le Grand, B., & Rossi, F. (2016). Mean absolute percentage error for regression models. *Neurocomputing*, 192, 38-48.

Delkic, Melina (December 12, 2017). What Gun Laws Passed in 2017? Trump Isn't Doing Anything About Gun Violence but States Are Stepping Up*, U.S. Newsweek*. Retrieved January 08, 2021 from https://www.newsweek.com/gun-laws-mass-shootings-2017-donald-trump-751884

Fleegler, E. W., Lee, L. K., Monuteaux, M. C., Hemenway, D. & Mannix, R. (2013). Firearm legislation and firearm-related fatalities in the United States. JAMA Internal Medicine, 173(9), 732–740.

Fowler, K. A., Dahlberg, L. L., Haileyesus, T. & Annest, J. L. (2015). Firearm injuries in the United States. *Preventive Medicine*, 79, 5–14.

Gabor, T. (2016). Confronting gun violence in America. Springer.

Gambino, Lauren (July 7, 2014). Gun violence flares up across US in bloody Fourth of July weekend, The Guardian. Online Retrieved January 10, 2020 from https://www.theguardian.com/world/2014/jul/07/gun-violence-fourth-of-july-weekend

Grinshteyn, E. & Hemenway, D. (2016). Violent death rates: the US compared with other high-income OECD countries, 2010. *The American Journal of Medicine*, 129(3), 266–273.

GVA - Gun Violence Archive. (2020). *Methodology*. Retrieved January 10, 2020 from: https://www.gunviolencearchive.org/

Harvey, A. C. & Peters, S. (1990). Estimation procedures for structural time series models. *Journal of Forecasting*, 9(2), 89–108.

Hastie, T. J., & Tibshirani, R. J. (1990). Generalized additive models (Vol. 43). CRC press.

Hemenway, D. & Miller, M. (2000). Firearm availability and homicide rates across 26 high-income countries. *Journal of Trauma and Acute Care Surgery*, 49(6), 985–988.

Ko, J. (2018). Gun Violence Data : GitHub repository. Retrieved January 10, 2020 from: https://github.com/jamesqo/gun-violence-data

Khair, U., Fahmi, H., Al Hakim, S., & Rahim, R. (2017, December). Forecasting error calculation with mean absolute deviation and mean absolute percentage error. In Journal of Physics: Conference Series (Vol. 930, No. 1, p. 012002). IOP Publishing.

Larsen, D. A., Lane, S., Jennings-Bey, T., Haygood-El, A., Brundage, K. & Rubinstein, R. A. (2017). Spatio-temporal patterns of gun violence in Syracuse, New York

Lee, L. K., Fleegler, E. W., Farrell, C., Avakame, E., Srinivasan, S., Hemenway, D. & Monuteaux, M. C. (2017). Firearm laws and firearm homicides: a systematic review. *JAMA Internal Medicine*, 177(1), 106–119.

Meloy, J. R., Hempel, A. G., Gray, B. T., Mohandie, K., Shiva, A. & Richards, T. C. (2004). A comparative analysis of North American adolescent and adult mass murderers. *Behavioral Sciences & the Law*, 22(3), 291–309.

Page, E. (2009). Men, masculinity and guns: Can we break the link. London: IANSA–Women's Network, No Date.

Robson, W. (June 17, 2019). The Math of Proph& - Future Vision - Medium. Future Vision. Retrieved January 10, 2020 from: https://medium.com/future-vision/the-math-of-prophet-46864fa9c55a

Taylor, S. J. & Letham, B. (2018). Forecasting at scale. *The American Statistician*, 72(1), 37–45.

Taylor, S. J. & Letham, B. (2020). Forecasting at scale (Webpage), Diagnostics, Retrieved January 10, 2021 from: https://facebook.github.io/prophet/docs/diagnostics.html

Wang, X. & Brown, D. E. (2012). The spatio-temporal modeling for criminal incidents. *Security Informatics*, 1(1), 1-12.

Assignment Data Science

IM939:Data Science Across Disciplines: Principles, Practice and Critique

Lukas Birkenmaier, u2085320

0. Loading Packages and Data

Load Packages and Data Source: https://github.com/jamesqo/gun-violence-data (https://github.com/jamesqo/gun-violence-data)

In [156]:

```
import json
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from fbprophet import Prophet
import altair as alt
```

In [157]:

```
df = pd.read_csv('stage3.csv')
```

1. Data Inspection and Visualization

1.1 General Data Inspection

In [158]:

```
df.dtypes
```

Out[158]:

```
incident_id                    int64
date                          object
state                         object
city_or_county                object
address                       object
n_killed                       int64
n_injured                      int64
incident_url                  object
source_url                    object
incident_url_fields_missing     bool
congressional_district       float64
gun_stolen                    object
gun_type                      object
incident_characteristics      object
latitude                     float64
location_description          object
longitude                    float64
n_guns_involved              float64
notes                         object
participant_age               object
participant_age_group         object
participant_gender            object
participant_name              object
participant_relationship      object
participant_status            object
participant_type              object
sources                       object
state_house_district         float64
state_senate_district        float64
dtype: object
```

In [159]:

```
df.shape
```

Out[159]:

```
(239677, 29)
```

In [160]:

```
df.head()
```

Out[160]:

	incident_id	date	state	city_or_county	address	n_killed	n_injured	
0	461105	2013-01-01	Pennsylvania	Mckeesport	1506 Versailles Avenue and Coursin Street	0	4	http://www.gu
1	460726	2013-01-01	California	Hawthorne	13500 block of Cerise Avenue	1	3	http://www.gu
2	478855	2013-01-01	Ohio	Lorain	1776 East 28th Street	1	3	http://www.gu
3	478925	2013-01-05	Colorado	Aurora	16000 block of East Ithaca Place	4	0	http://www.gu
4	478959	2013-01-07	North Carolina	Greensboro	307 Mourning Dove Terrace	2	2	http://www.gu

5 rows × 29 columns

In [161]:

```
df.isnull().sum()
```

Out[161]:

```
incident_id                      0
date                             0
state                            0
city_or_county                   0
address                      16497
n_killed                         0
n_injured                        0
incident_url                     0
source_url                     468
incident_url_fields_missing      0
congressional_district       11944
gun_stolen                   99498
gun_type                     99451
incident_characteristics       326
latitude                      7923
location_description        197588
longitude                     7923
n_guns_involved              99451
notes                        81017
participant_age              92298
participant_age_group        42119
participant_gender           36362
participant_name            122253
participant_relationship    223903
participant_status           27626
participant_type             24863
sources                        609
state_house_district         38772
state_senate_district        32335
dtype: int64
```

1.2 Exploring Time Related Trends of Gun Violence

In [162]:

```
df['datetime'] = pd.to_datetime(df['date'])
```

First: Extracting the number of

- incidents (1 incident = 1 observation)
- people injured
- people killed

and storing them in a new dataframe 'df_trend'

In [163]:

```
df_kills = df['n_killed'].groupby(by = df['datetime']).sum()  #Matrix of Absolut
e Cases
df_harmed = df['n_injured'].groupby(by = df['datetime']).sum()  #Matrix of Absol
ute Cases
df_cases = df['incident_id'].groupby(by = df['datetime']).count()  #Matrix of Ab
solute Cases

df_cases = pd.DataFrame(df_cases)
df_cases = df_cases.reset_index()

df_harmed = pd.DataFrame(df_harmed)
df_harmed = df_harmed.reset_index()

df_kills = pd.DataFrame(df_kills)
df_kills = df_kills.reset_index()


df_trend = pd.concat([df_kills,df_harmed['n_injured'],df_cases['incident_id']],a
xis=1)
df_trend = df_trend.rename(columns={"n_killed": "Total Number of People killed",
"n_injured": "Total Number of People injured",'incident_id':'Total Number of Inc
idents'})
```

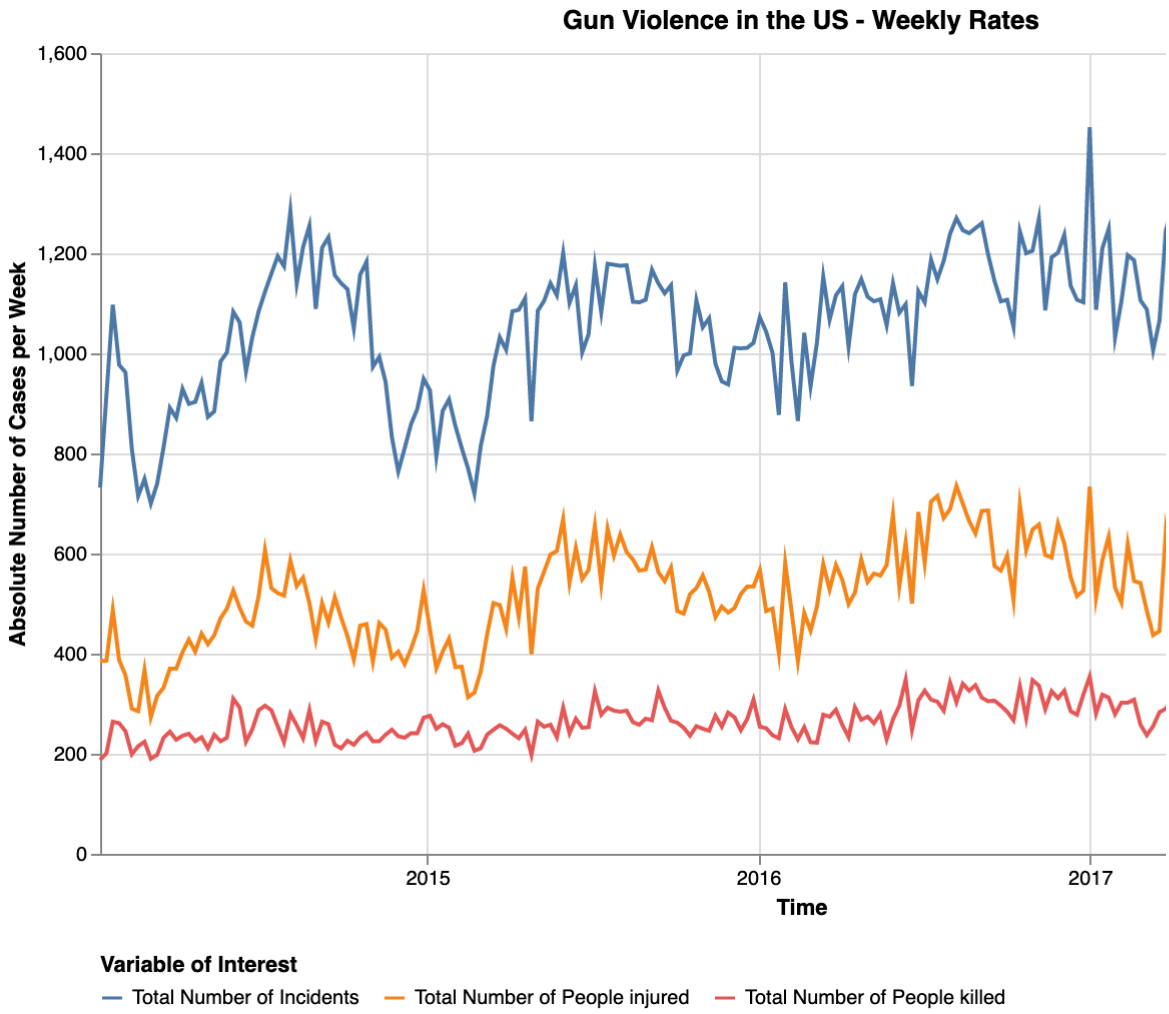
Picture on a weekly basis

In [164]:

```
df_trend2 = pd.melt(df_trend, id_vars = ['datetime'] )
df_weekly = df_trend2.set_index('datetime')
df_weekly = df_weekly.groupby('variable').resample('W').sum()
df_weekly = df_weekly.reset_index()
df_weekly = df_weekly[df_weekly['value']!= 0]

alt.Chart(df_weekly[df_weekly['datetime'] >= '01-01-2014']).mark_line().encode(
    x=alt.X('datetime:T', title = 'Time', axis = alt.Axis(format='%Y' , tickCoun
t= 5),
 ),
    y= alt.Y('value:Q', title = 'Absolute Number of Cases per Week'),
    color= alt.Color('variable', title = 'Variable of Interest', )
).properties(width=700, height = 400,title = 'Gun Violence in the US - Weekly Ra
tes').configure_view(
    stroke='transparent'
).configure_legend(orient = 'bottom')
```

Out[164]:



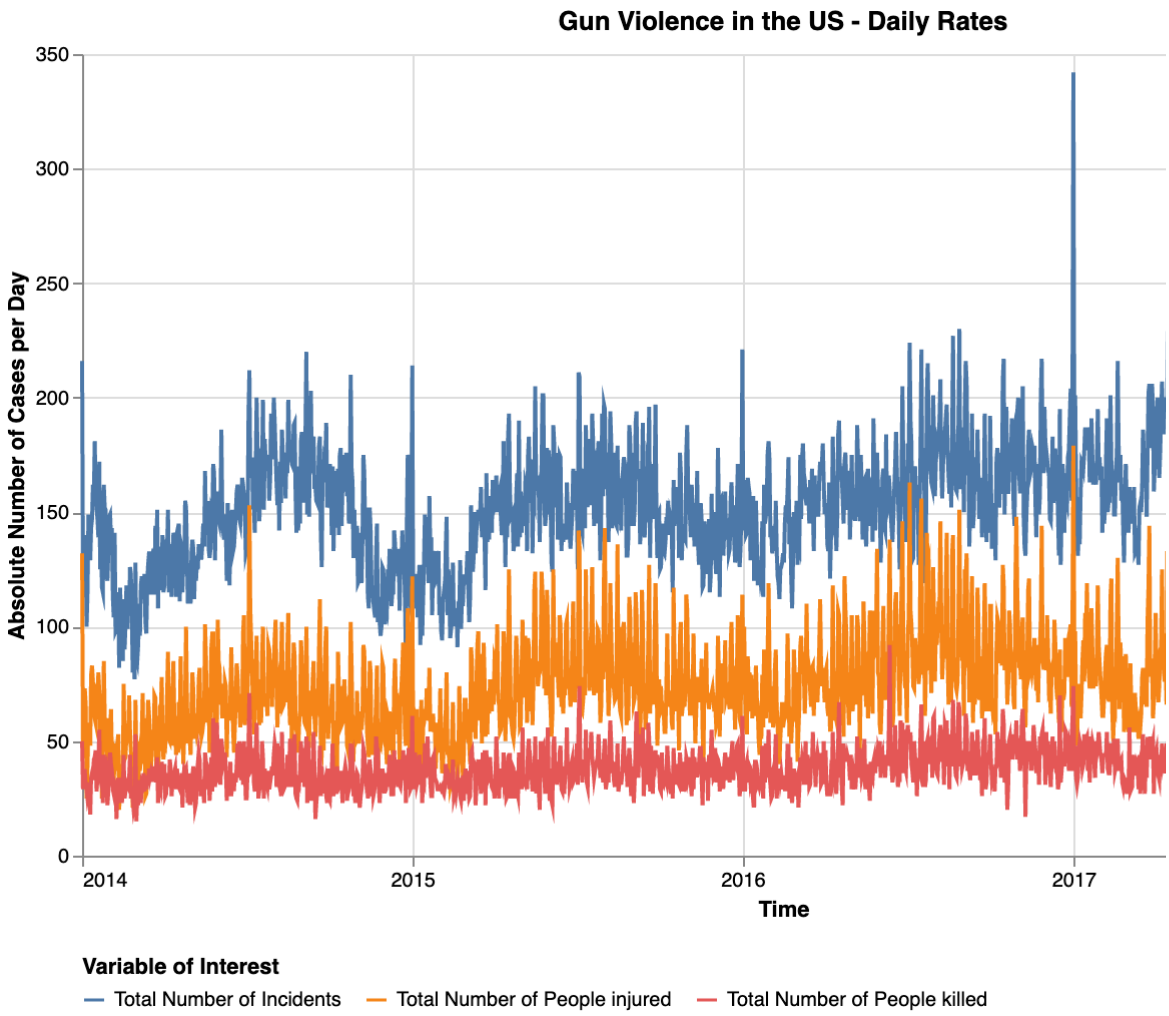
Picture on a daily basis

In [165]:

```
df_trend2 = pd.melt(df_trend, id_vars = ['datetime'] )
df_weekly = df_trend2.set_index('datetime')
df_weekly = df_weekly.groupby('variable').resample('D').sum()
df_weekly = df_weekly.reset_index()
df_weekly = df_weekly[df_weekly['value']!= 0]

alt.Chart(df_weekly[df_weekly['datetime'] >= '01-01-2014']).mark_line().encode(
    x=alt.X('datetime:T', title = 'Time', axis = alt.Axis(format='%Y' , tickCount= 5),
 ),
    y= alt.Y('value:Q', title = 'Absolute Number of Cases per Day'),
    color= alt.Color('variable', title = 'Variable of Interest', )
).properties(width=700, height = 400, title = 'Gun Violence in the US - Daily Rates').configure_view(
    stroke='transparent'
).configure_legend(orient = 'bottom')
```

Out[165]:



This looks better.

However, there are a lot of datapoints (since there are no days where the total amount of incidents is below 50). Maybe I should narrow down my plot and inspect aggregated levels on year, month and day basis

In [166]:

```
df['weekday'] = df['datetime'].dt.day_name()
df['month'] = df['datetime'].dt.month_name()
df['year'] = df['datetime'].dt.year
df['weekday'] = df['weekday'].astype('object')
df['month'] = df['month'].astype('object')
df['year'] = df['year'].astype('object')
```

A. Changes over Years

In [199]:

```
df['number'] = 1
df_years = df.groupby('year').agg('sum')[['n_killed','n_injured','number']]
df_years = df_years.rename(index = str, columns={"n_killed":"People Killed", "n_
injured":"People Injured", 'number':'Number of Incidents'})
df_years = df_years.reset_index()
df_years = pd.melt(df_years, id_vars = ['year'], value_vars = ['People Killed',
'People Injured','Number of Incidents'])

df_months = df.groupby('month').agg('sum')[['n_killed','n_injured','number']]
df_months = df_months.rename(index = str, columns={"n_killed":"People Killed",
"n_injured":"People Injured", 'number':'Number of Incidents'})
df_months = df_months.reset_index()
df_months = pd.melt(df_months, id_vars = ['month'], value_vars = ['People Kille
d', 'People Injured','Number of Incidents'])

df_days = df.groupby('weekday').agg('sum')[['n_killed','n_injured','number']]
df_days = df_days.rename(index = str, columns={"n_killed":"People Killed", "n_in
jured":"People Injured", 'number':'Number of Incidents'})
df_days = df_days.reset_index()
df_days = pd.melt(df_days, id_vars = ['weekday'], value_vars = ['People Killed',
'People Injured','Number of Incidents'])
```

In [200]:

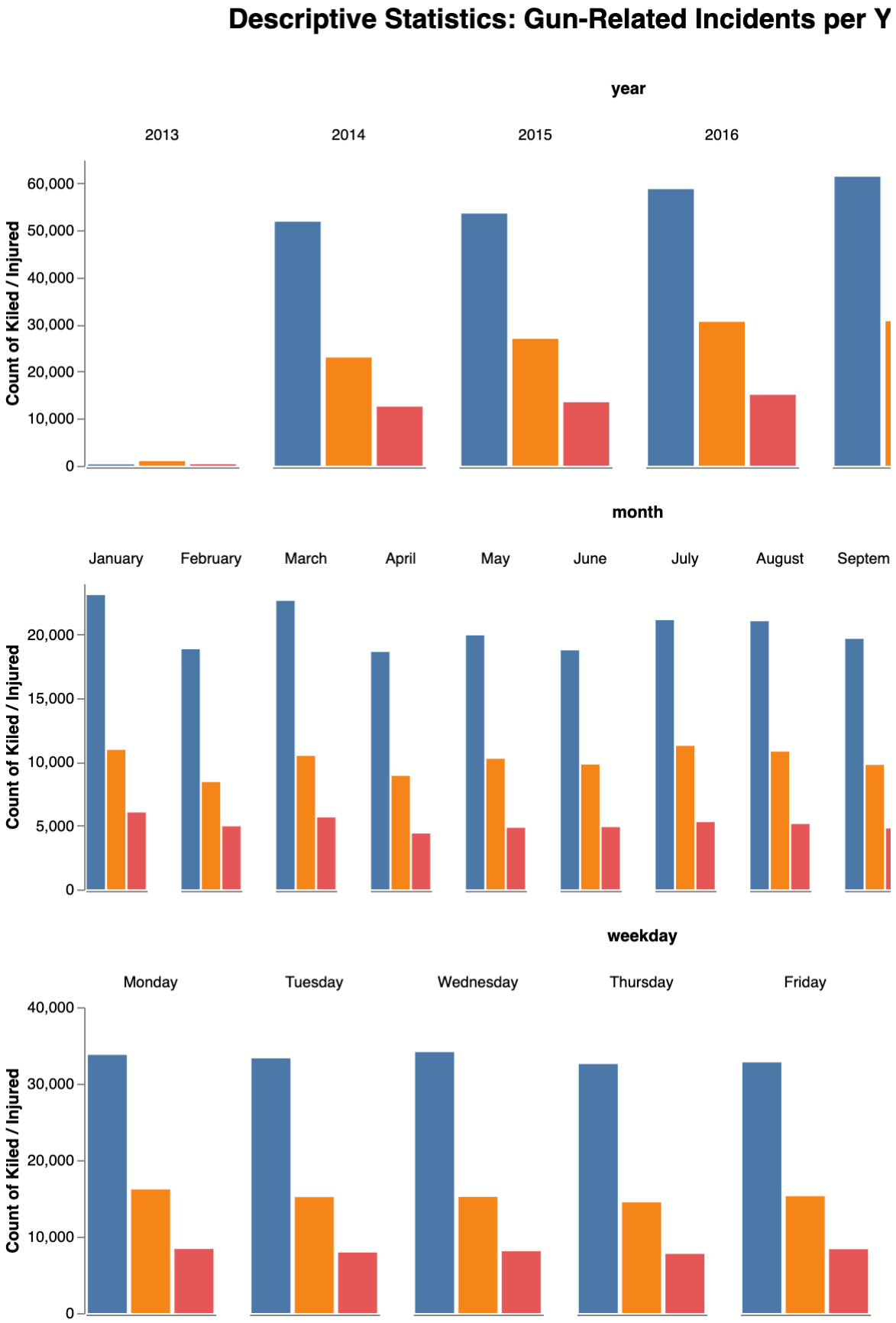
```
a = alt.Chart(df_years).mark_bar().encode(
    x= alt.X('variable:O', axis=alt.Axis(labels=False, ticks=False, title= None
)),
    y= alt.Y('value:Q', axis=alt.Axis(grid=False, title='Count of Kiled / Injure
d')),
    color='variable:N',
    column=alt.Column('year:N')
).properties(
    width=100, height = 200)

b= alt.Chart(df_months).mark_bar().encode(
    x= alt.X('variable:O', axis=alt.Axis(labels=False, ticks=False, title= None
)),
    y= alt.Y('value:Q', axis=alt.Axis(grid=False, title='Count of Kiled / Injure
d')),
    color='variable:N',
    column=alt.Column('month:N', sort = ['January','February','March','April','M
ay','June','July','August','September','October','November','December'])
).properties(
    width=40, height = 200)

c = alt.Chart(df_days).mark_bar().encode(
    x= alt.X('variable:O', axis=alt.Axis(labels=False, ticks=False, title= None
)),
    y= alt.Y('value:Q', axis=alt.Axis(grid=False, title='Count of Kiled / Injure
d')),
    color='variable:N',
    column=alt.Column('weekday:N', sort = ['Monday','Tuesday','Wednesday','Thurs
day','Friday','Saturday','Sunday'])
).properties(width=85, height = 200)

alt.vconcat(a, b,c).properties(title = 'Descriptive Statistics: Gun-Related Inci
dents per Year, Month and Week').configure_view(stroke='transparent').configure_
title(anchor='middle', fontSize=18, offset = 30)
```

Out[200]:


Descriptive Statistics: Gun-Related Incidents per Y
year
2013
2014
2015
2016
Count of Kiled / Injured
60,000
50,000
40,000
30,000
20,000
10,000
0
month
January
February
March
April
May
June
July
August
Septem
Count of Kiled / Injured
20,000
15,000
10,000
5,000
0
weekday
Monday
Tuesday
Wednesday
Thursday
Friday
Count of Kiled / Injured
40,000
30,000
20,000
10,000
0

Changes over Days

1.2.1 Identifying Holidays

In [172]:

```
df_freq = pd.crosstab(df['date'], columns='count')
df_freq = pd.DataFrame(df_freq)
df_freq = df_freq.reset_index()
df_freq['date'] = pd.to_datetime(df_freq['date'])
df_freq = df_freq[df_freq['date'] >= '01-01-2014']
df_freq_asc = df_freq[['count','date']].sort_values(by = ['count'],ascending = T
rue)
df_freq_des = df_freq[['count','date']].sort_values(by = ['count'],ascending = F
alse)
```

In [174]:

```
df_freq_asc[0:20]
```

Out[174]:

col_0	count	date
235	77	2014-02-28
233	80	2014-02-26
218	82	2014-02-11
222	85	2014-02-15
238	85	2014-03-03
535	87	2014-12-25
237	87	2014-03-02
224	90	2014-02-17
592	91	2015-02-20
239	92	2014-03-04
551	92	2015-01-10
575	94	2015-02-03
584	95	2015-02-12
507	96	2014-11-27
553	96	2015-01-12
241	96	2014-03-06
248	97	2014-03-13
214	98	2014-02-07
574	98	2015-02-02
215	99	2014-02-08

In [175]:

```
df_freq_des[0:20]
```

Out[175]:

col_0	count	date
1271	342	2017-01-01
1454	248	2017-07-04
1635	242	2018-01-01
1417	242	2017-05-28
1145	230	2016-08-28
1375	229	2017-04-16
1138	227	2016-08-21
1090	224	2016-07-04
1381	222	2017-04-22
906	221	2016-01-01
1103	221	2016-07-17
1563	220	2017-10-21
425	220	2014-09-06
1418	217	2017-05-29
1416	217	2017-05-27
1194	217	2016-10-16
1236	217	2016-11-27
1396	216	2017-05-07
1152	216	2016-09-04
177	216	2014-01-01

There is a heavy spike on 01.01.2017 with a very high number of 342 cases

To extract regular occuring (holiday) events, I will take the subset of the 200 days with the highest amount of cases and look which repeat every year

In [176]:

```
df_freq = df_freq.reset_index()
df_freq['datetime'] = pd.to_datetime(df_freq['date'])# Making sure that there is
a datetime format
df_freq['datetstr'] = df_freq['datetime'].apply(lambda x: x.strftime('%Y-%m-%d'
)) #Generating string from datetime-format
df_freq['Month-day'] = df_freq['datetstr'].str[5:] #Removing the year

df_freq_200 = df_freq.sort_values(by = 'count', ascending = False)[0:200] ##Only
extracting the 200 days with the higesht number of cases
pd.crosstab(df_freq_200['Month-day'], columns='count').sort_values(by = 'count',
ascending = False)
```

Out[176]:

col_0	count
Month-day	
01-01	5
07-30	4
07-05	4
07-04	4
09-05	3
...	...
05-14	1
05-13	1
05-10	1
05-08	1
12-31	1

150 rows × 1 columns

Ok, here I can see that there are several holidays with an increase in cases:

- January 01 (New Year)
- July 04 (Indepence Day) (and 05.07)
- September 05 (Maybe Labor Day?)

Looking at the dates for Christmas

In [177]:

```
print('Counts for Christmas\n')
print(df_freq[df_freq['Month-day']== '12-24'])
print('\nCounts for Valentines Day\n')
print(df_freq[df_freq['Month-day']== '02-14'])
print('\n...')
```

```
Counts for Christmas

col_0  index       date  count   datetime    datetstr Month-day
357      534 2014-12-24    111 2014-12-24  2014-12-24     12-24
721      898 2015-12-24    128 2015-12-24  2015-12-24     12-24
1086    1263 2016-12-24    156 2016-12-24  2016-12-24     12-24
1450    1627 2017-12-24    133 2017-12-24  2017-12-24     12-24

Counts for Valentines Day

col_0  index       date  count   datetime    datetstr Month-day
44       221 2014-02-14    101 2014-02-14  2014-02-14     02-14
409      586 2015-02-14    104 2015-02-14  2015-02-14     02-14
773      950 2016-02-14    128 2016-02-14  2016-02-14     02-14
1137    1314 2017-02-14    168 2017-02-14  2017-02-14     02-14
1502    1679 2018-02-14    141 2018-02-14  2018-02-14     02-14

...
```

2. Modellinging

Sources:

- https://www.digitalocean.com/community/tutorials/a-guide-to-time-series-forecasting-with-prophet-in-python-3#step-2-%E2%80%94-import-packages-and-load-data (https://www.digitalocean.com/community/tutorials/a-guide-to-time-series-forecasting-with-prophet-in-python-3#step-2-%E2%80%94-import-packages-and-load-data)
- https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html#built-in-country-holidays (https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html#built-in-country-holidays) (official source)

2.1 Preprocessing

Reducing to the time period with valid data

In [178]:

```
df_prophet = df_freq[['date','count']]
df_prophet = df_prophet.rename(columns={'date': 'ds',
                        'count': 'y'})
df_prophet = df_prophet.loc[(df_prophet.ds >= '2014-01-02')] #From here, the dat
a starts to look well
df_prophet = df_prophet[df_prophet.ds.notnull()]
```

2.2 Modelling Number of Cases

2.2.1 Model 1: Baseline Model

In [179]:

```
my_model_base = Prophet(interval_width=0.95, daily_seasonality= False) #Specific
ation of the Model
my_model_base.fit(df_prophet) #Fitting Model to data
future_dates = my_model_base.make_future_dataframe(periods=720) #Make future dat
aset for the next 720 days
forecast = my_model_base.predict(future_dates) #Predict Y for the future dataset
```

2.2.2 Model 2: Including Holidays

Including Public Holidays

There are two version to add holidays.

- one can include the dates manually (2.2.2.1)
- one can use the Built-in Country Holidays https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html#specifying-custom-seasonalities (https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html#specifying-custom-seasonalities) (2.2.2.2)

In this appraoch, I will focus on the automatic Inclusion of Holidays, which are:

In [180]:

```
my_model_holidays = Prophet(interval_width=0.95, daily_seasonality= False)
my_model_holidays.add_country_holidays(country_name='US')
my_model_holidays.fit(df_prophet)
future_dates = my_model_holidays.make_future_dataframe(periods=720) #Make predic
tions for the next 36 months
forecast_holidays = my_model_holidays.predict(future_dates)
```

2.2.3 Model 3: Adding regressors (not included in final analysis, because no data on daily level was available)

my_model_regressors = Prophet(interval_width=0.95, daily_seasonality= False)
my_model_regressors.add_country_holidays(country_name='US')

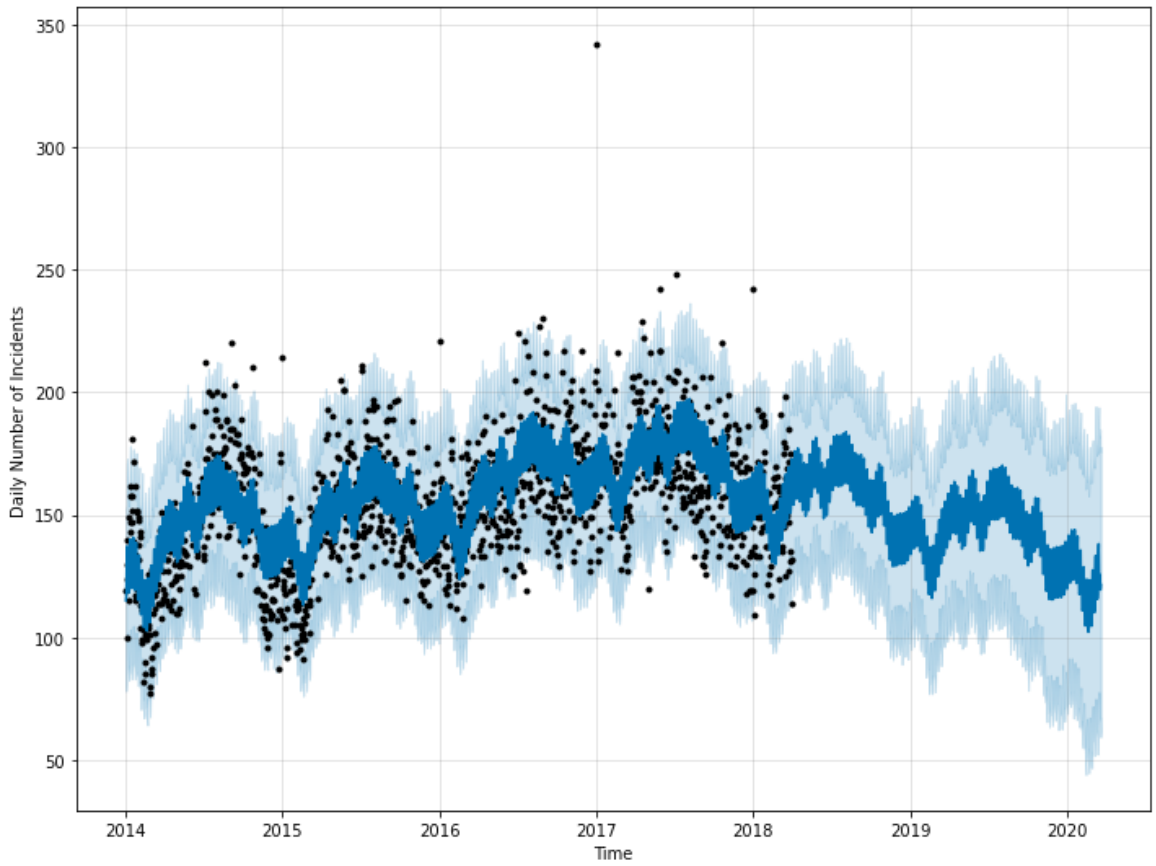
my_model_regressors.fit(df_prophet) future_dates_regressors =
my_model_regressors.make_future_dataframe(periods=720) #Make predictions for the next 36 months
forecast_regressor = my_model_regressors.predict(future_dates_regressors)

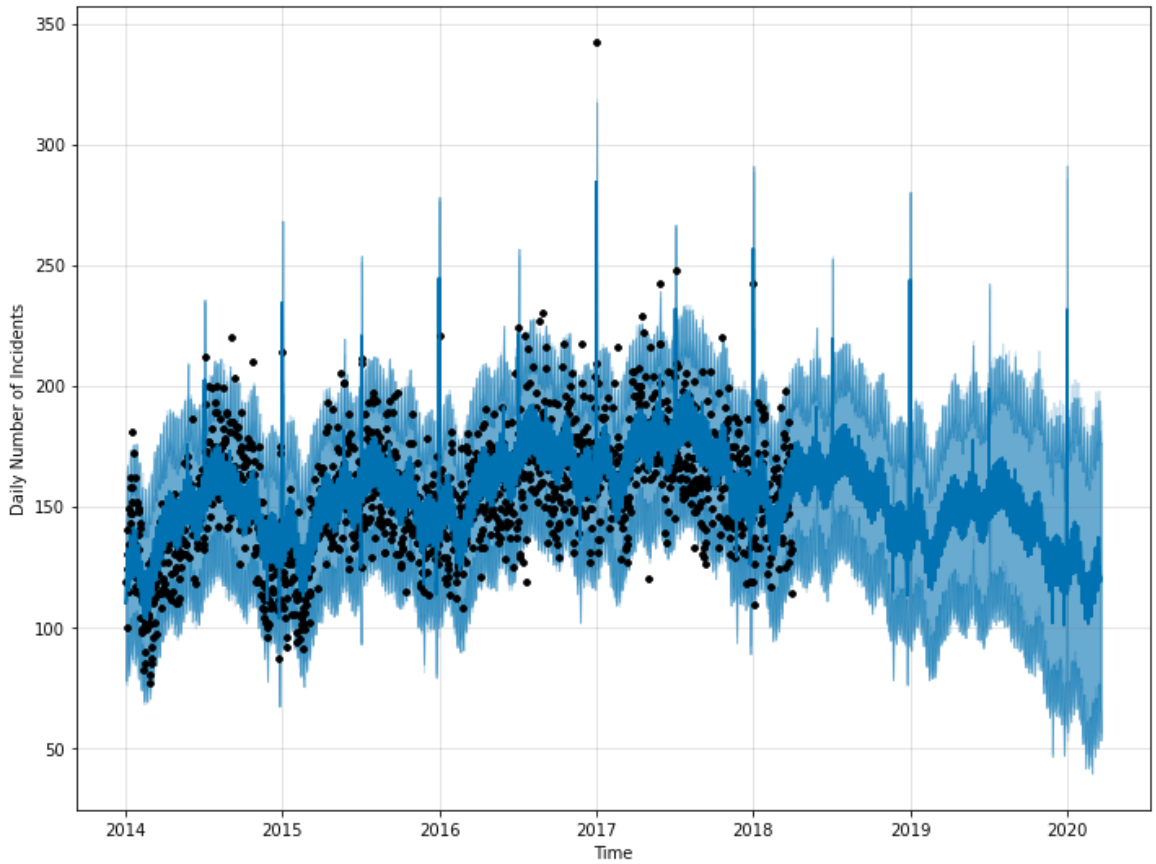
Plotting

In [181]:

```
plot = my_model_holidays.plot(forecast_holidays, ax=ax2, xlabel= 'Time', ylabel= 'Daily Number of Incidents')
plot
```

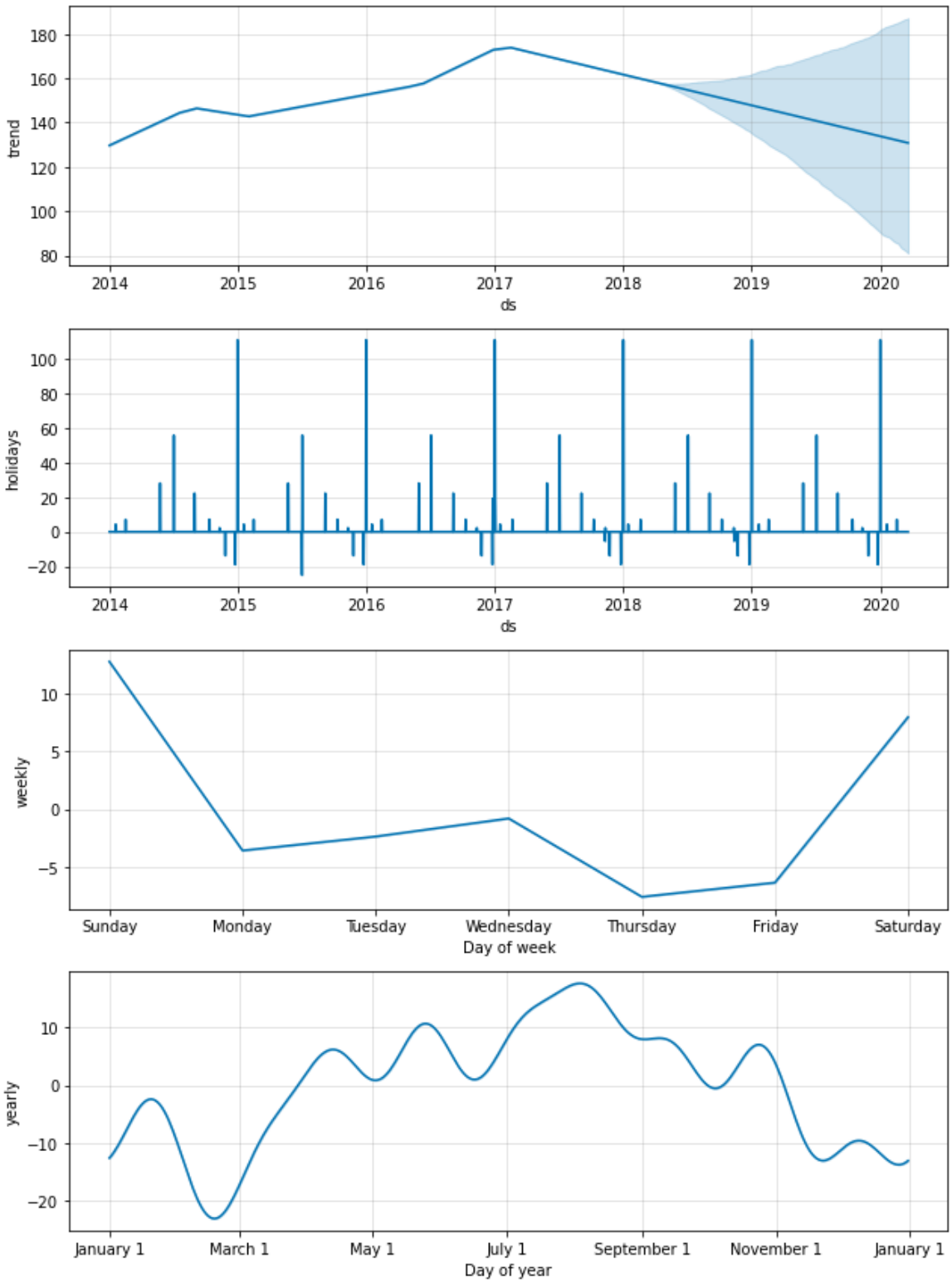
Out[181]:





In [182]:

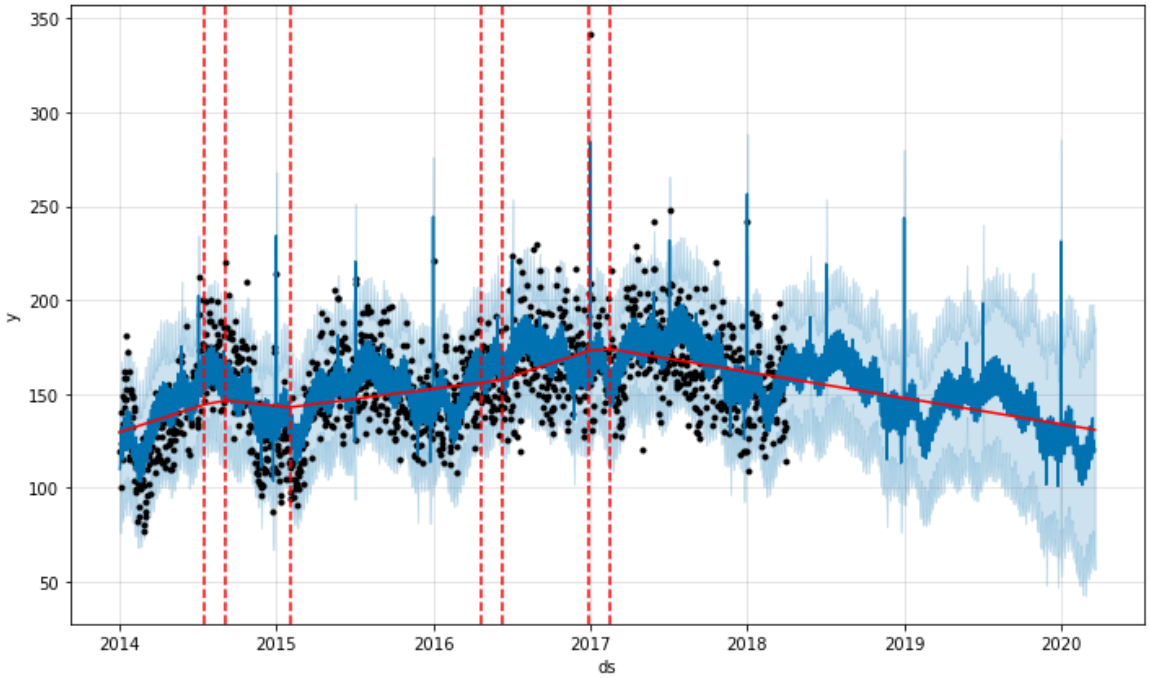
```
my_model_holidays.plot_components(forecast_holidays).savefig('prophetplot_components.svg')
```



Final Plot: Inclusion of Trend

In [183]:

```
from fbprophet.plot import add_changepoints_to_plot
fig = my_model_holidays.plot(forecast_holidays)
a = add_changepoints_to_plot(fig.gca(), my_model_holidays, forecast_holidays)
```



In [ ]:

```
fig.savefig('prophetplot.svg')
```

Showing possible changes in the trends (Source: https://facebook.github.io/prophet/docs/trend_changepoints.html#automatic-changepoint-detection-in-prophet (https://facebook.github.io/prophet/docs/trend_changepoints.html#automatic-changepoint-detection-in-prophet))

There seems to be a big change in the trend around the start of 2017. This is something, which should be further investigated, since it seems to have had a significant negative impact on the trend

Model Diagnostics

Extracting Performance Metrices

In [ ]:

```
from fbprophet.diagnostics import cross_validation
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric

df_cv_base = cross_validation(my_model_base, initial='750 days', period='90 day
s', horizon = '365 days')
df_p_base = performance_metrics(df_cv_base)

df_cv_holidays = cross_validation(my_model_holidays, initial='750 days', period=
'90 days', horizon = '365 days')
df_p_holidays = performance_metrics(df_cv_holidays)
```

Comparing

In [185]:

```
df_p_holidays.loc[df_p_holidays['horizon'].isin(['37 days','60 days','90 days',
'180 days','360 days'])]
```

Out[185]:

	horizon	mse	rmse	mae	mape	mdape	coverage
0	37 days	390.062305	19.749995	15.158324	0.092331	0.069699	0.897802
23	60 days	664.609389	25.780019	20.520820	0.122603	0.099171	0.769231
53	90 days	812.722157	28.508282	23.731528	0.146502	0.136602	0.743956
143	180 days	1425.509282	37.755917	31.450476	0.195463	0.156721	0.561538
323	360 days	1141.881183	33.791732	27.547273	0.182575	0.152165	0.654945

Manually adding Holidays (not used in final analysis

New_Years = pd.DataFrame({ #01. January 'holiday': 'New Year', 'ds': pd.to_datetime(['2014-01-01', '2015-01-01', '2016-01-01', '2017-01-01', '2018-01-01', '2019-01-01', '2020-01-01', '2021-01-01', '2022-01-01', '2023-01-01', '2024-01-01', '2025-01-01',]) })

Indepence = pd.DataFrame({ #04. July 'holiday': 'Indepence Day', 'ds': pd.to_datetime(['2014-07-04', '2015-07-04', '2016-07-04', '2017-07-04', '2018-07-04', '2019-07-04', '2020-07-04', '2021-07-04', '2022-07-04', '2023-07-04', '2024-07-04', '2025-07-04']), })

holidays_for_prophet = pd.concat((New_Years, Indepence))

my_model_holidays_manually = Prophet(interval_width=0.95, yearly_seasonality= True, weekly_seasonality= True,holidays= holidays_for_prophet) my_model_holidays_manually.fit(df_prophet) future_dates = my_model_holidays_manually.make_future_dataframe(periods=720) #Make predictions for the next 36 months forecast_holidays_manually = my_model_holidays_manually.predict(future_dates)