Lightning Strikes EDA Project

October 21, 2025

[ ]:

```
# Import libraries and packages
import pandas as pd
import numpy as np
import seaborn as sns
import datetime
from matplotlib import pyplot as plt
```

[2]:

```
# Read in the 2018 data.
df = pd.read_csv('eda_structuring_with_python_dataset1.csv')
df.head()
```

[2]:

```
         date  number_of_strikes center_point_geom
0  2018-01-03                194     POINT(-75 27)
1  2018-01-03                 41   POINT(-78.4 29)
2  2018-01-03                 33   POINT(-73.9 27)
3  2018-01-03                 38   POINT(-73.8 27)
4  2018-01-03                 92     POINT(-79 28)
```

[3]:

```
# Convert the `date` column to datetime.
df['date'] = pd.to_datetime(df['date'])
```

[4]:

```
# Returns (Rows, Col)
df.shape
```

[4]:

```
(3401012, 3)
```

[5]:

```
# Check for duplicates - No dupicates found
df.drop_duplicates().shape
```

[5]:

```
(3401012, 3)
```

[6]:

```
# Sort by number of strikes in descending order.
df.sort_values(by='number_of_strikes', ascending=False).head(10)
```

[6]:

```
             date  number_of_strikes  center_point_geom
302758 2018-08-20               2211  POINT(-92.5 35.5)
278383 2018-08-16               2142  POINT(-96.1 36.1)
```

```
280830 2018-08-17                2061  POINT(-90.2 36.1)
280453 2018-08-17                2031  POINT(-89.9 35.9)
278382 2018-08-16                1902  POINT(-96.2 36.1)
11517  2018-02-10                1899  POINT(-95.5 28.1)
277506 2018-08-16                1878  POINT(-89.7 31.5)
24906  2018-02-25                1833  POINT(-98.7 28.9)
284320 2018-08-17                1767    POINT(-90.1 36)
24825  2018-02-25                1741      POINT(-98 29)
```

```
[7]: # Identify the locations that appear most in the dataset.
     df.center_point_geom.value_counts()
```

```
[7]: POINT(-81.5 22.5)     108
     POINT(-84.1 22.4)     108
     POINT(-82.5 22.9)     107
     POINT(-82.7 22.9)     107
     POINT(-82.5 22.8)     106
                          …
     POINT(-119.3 35.1)      1
     POINT(-119.3 35)        1
     POINT(-119.6 35.6)      1
     POINT(-119.4 35.6)      1
     POINT(-58.5 45.3)       1
     Name: center_point_geom, Length: 170855, dtype: int64
```

```
[8]: # Identify the top 20 locations with most days of lightning.
     df.center_point_geom.value_counts()[:20].rename_axis('unique_values').
      ↪reset_index(name='counts').style.background_gradient()
```

```
[8]: <pandas.io.formats.style.Styler at 0x71c9c02e5b90>
```

```
[9]: # Create two new columns.
     df['week'] = df.date.dt.isocalendar().week
     df['weekday'] = df.date.dt.day_name()
     df.head()
```

```
[9]:         date  number_of_strikes center_point_geom  week    weekday
     0 2018-01-03                194     POINT(-75 27)     1  Wednesday
     1 2018-01-03                 41   POINT(-78.4 29)     1  Wednesday
     2 2018-01-03                 33   POINT(-73.9 27)     1  Wednesday
     3 2018-01-03                 38   POINT(-73.8 27)     1  Wednesday
     4 2018-01-03                 92     POINT(-79 28)     1  Wednesday
```

```
[10]: # Calculate the mean count of lightning strikes for each weekday.
      df[['weekday','number_of_strikes']].groupby(['weekday']).mean()
```

[10]:

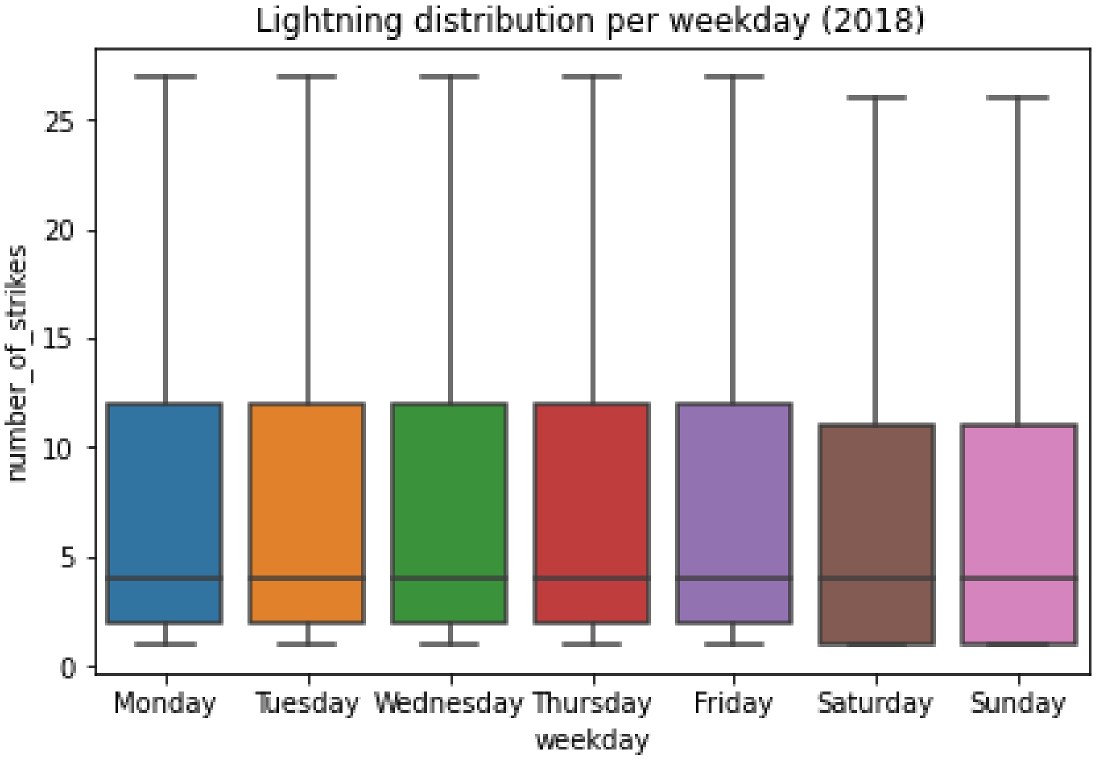
```
            number_of_strikes
weekday
Friday              13.349972
Monday              13.152804
Saturday            12.732694
Sunday              12.324717
Thursday            13.240594
Tuesday             13.813599
Wednesday           13.224568
```

[11]:

```
# Define order of days for the plot.
weekday_order = ['Monday','Tuesday', 'Wednesday',␣
 ↪'Thursday','Friday','Saturday','Sunday']
```

[20]:

```
# Create boxplots of strike counts for each day of week.
g = sns.boxplot(data=df,
            x='weekday',
            y='number_of_strikes',
            order=weekday_order,
            showfliers=False);
# Adjust layout spacing
plt.tight_layout(pad=1.0)
# Set Title
g.set_title('Lightning distribution per weekday (2018)');
```



[21]:

```python
# Import 2016-2017 data
df_2 = pd.read_csv('eda_structuring_with_python_dataset2.csv')
df_2.head()
```

[21]:

```
         date  number_of_strikes  center_point_geom
0  2016-01-04                 55  POINT(-83.2 21.1)
1  2016-01-04                 33  POINT(-83.1 21.1)
2  2016-01-05                 46  POINT(-77.5 22.1)
3  2016-01-05                 28  POINT(-76.8 22.3)
4  2016-01-05                 28    POINT(-77 22.1)
```

[22]:

```python
# Convert `date` column to datetime.
df_2['date'] = pd.to_datetime(df_2['date'])
```

[23]:

```python
# Create a new dataframe combining 2016-2017 data with 2018 data.
union_df = pd.concat([df.drop(['weekday','week'],axis=1), df_2],␣
 ↪ignore_index=True)
union_df.head()
```

[23]:

```
        date  number_of_strikes center_point_geom
0 2018-01-03                194     POINT(-75 27)
1 2018-01-03                 41   POINT(-78.4 29)
2 2018-01-03                 33   POINT(-73.9 27)
3 2018-01-03                 38   POINT(-73.8 27)
4 2018-01-03                 92     POINT(-79 28)
```

[24]:

```python
# Add 3 new columns.
union_df['year'] = union_df.date.dt.year
union_df['month'] = union_df.date.dt.month
union_df['month_text'] = union_df.date.dt.month_name()
union_df.head()
```

[24]:

```
        date  number_of_strikes center_point_geom  year  month month_text
0 2018-01-03                194     POINT(-75 27)  2018      1    January
1 2018-01-03                 41   POINT(-78.4 29)  2018      1    January
2 2018-01-03                 33   POINT(-73.9 27)  2018      1    January
3 2018-01-03                 38   POINT(-73.8 27)  2018      1    January
4 2018-01-03                 92     POINT(-79 28)  2018      1    January
```

[25]:

```python
# Calculate total number of strikes per year
union_df[['year','number_of_strikes']].groupby(['year']).sum()
```

[25]:

```
      number_of_strikes
year
2016           41582229
```

```
2017          35095195
2018          44600989
```

[28]:

```
# Calculate total lightning strikes for each month of each year.
lightning_by_month = union_df.groupby(['month_text','year']).agg(
    number_of_strikes = pd.NamedAgg(column='number_of_strikes',aggfunc=sum)).
 ↪reset_index()

lightning_by_month.head()
```

[28]:

```
  month_text  year  number_of_strikes
0      April  2016            2636427
1      April  2017            3819075
2      April  2018            1524339
3     August  2016            7250442
4     August  2017            6021702
```

[29]:

```
# Calculate total lightning strikes for each year.
lightning_by_year = union_df.groupby(['year']).agg(
  year_strikes = pd.NamedAgg(column='number_of_strikes',aggfunc=sum)
).reset_index()

lightning_by_year.head()
```

[29]:

```
   year  year_strikes
0  2016      41582229
1  2017      35095195
2  2018      44600989
```

[30]:

```
# Combine `lightning_by_month` and `lightning_by_year` dataframes into single␣
 ↪dataframe.
percentage_lightning = lightning_by_month.merge(lightning_by_year,on='year')
percentage_lightning.head()
```

[30]:

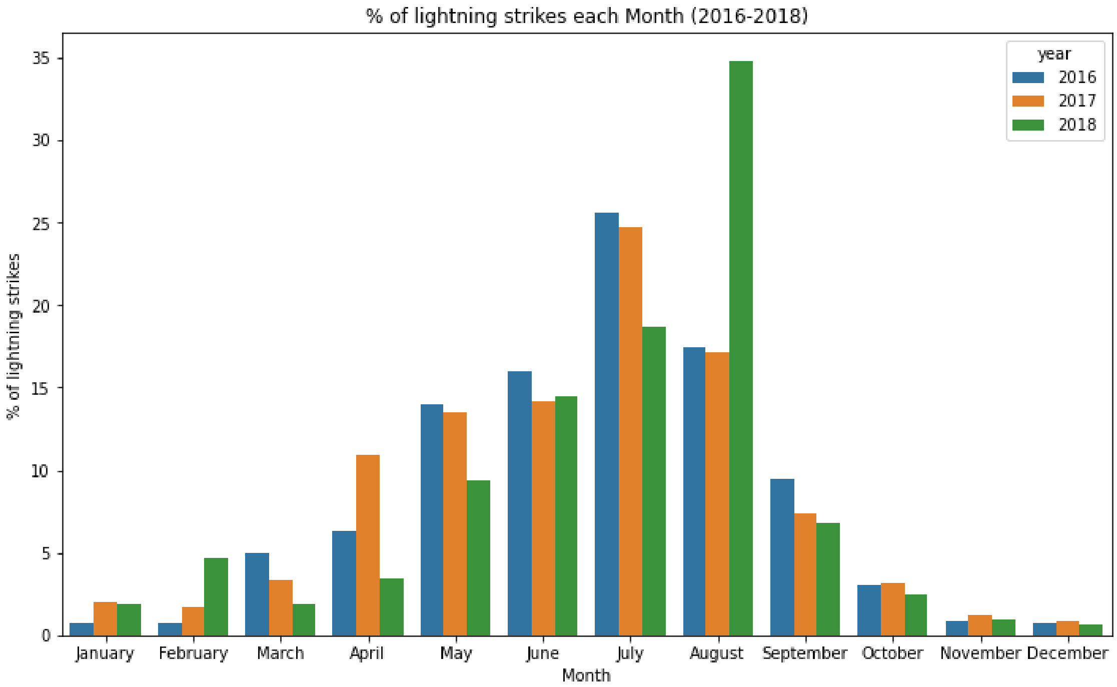
```
  month_text  year  number_of_strikes  year_strikes
0      April  2016            2636427      41582229
1     August  2016            7250442      41582229
2   December  2016             316450      41582229
3   February  2016             312676      41582229
4    January  2016             313595      41582229
```

[31]:

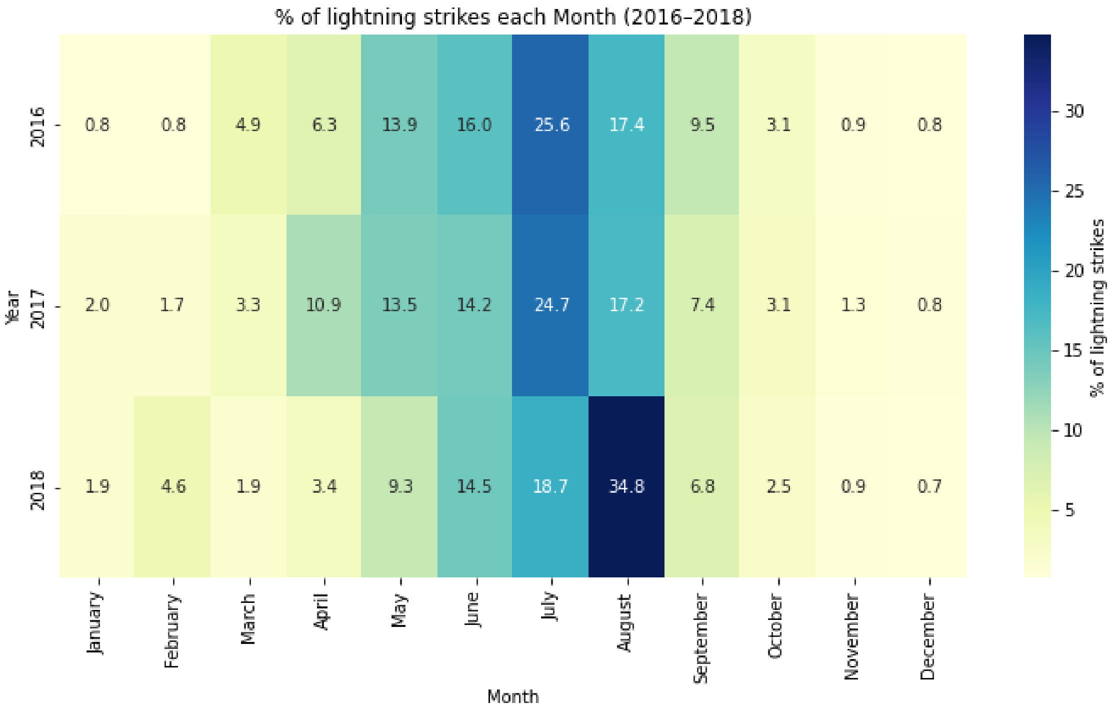
```
# Create new `percentage_lightning_per_month` column.
percentage_lightning['percentage_lightning_per_month'] = (percentage_lightning.
 ↪number_of_strikes/
                                                          percentage_lightning.
 ↪year_strikes * 100.0)
percentage_lightning.head()
```

[31]:
```
  month_text  year  number_of_strikes  year_strikes  \
0      April  2016            2636427      41582229
1     August  2016            7250442      41582229
2   December  2016             316450      41582229
3   February  2016             312676      41582229
4    January  2016             313595      41582229

   percentage_lightning_per_month
0                        6.340273
1                       17.436396
2                        0.761022
3                        0.751946
4                        0.754156
```

[33]:
```
plt.figure(figsize=(10,6));

month_order = ['January', 'February', 'March', 'April', 'May', 'June',
               'July', 'August', 'September', 'October', 'November', 'December']

sns.barplot(
    data = percentage_lightning,
    x = 'month_text',
    y = 'percentage_lightning_per_month',
    hue = 'year',
    order = month_order );
plt.xlabel("Month");
plt.ylabel("% of lightning strikes");
# Adjust layout spacing
plt.tight_layout(pad=1.0),
plt.title("% of lightning strikes each Month (2016-2018)");
```

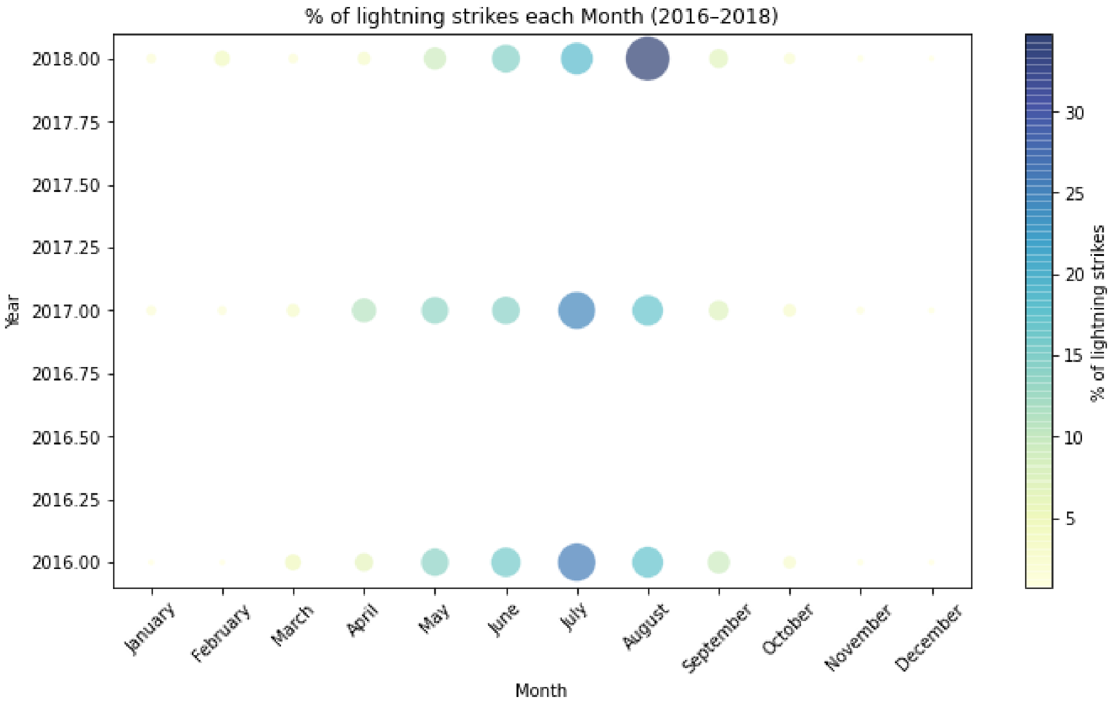


[34]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Ensure months are ordered correctly
month_order = ['January', 'February', 'March', 'April', 'May', 'June',
               'July', 'August', 'September', 'October', 'November', 'December']

# Pivot the data to create a matrix for the heatmap
heatmap_data = percentage_lightning.pivot_table(
    index='year',
    columns='month_text',
    values='percentage_lightning_per_month'
)

# Reorder the columns to match month_order
heatmap_data = heatmap_data[month_order]

# Plot the heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(heatmap_data, annot=True, fmt=".1f", cmap="YlGnBu",␣
 ↪cbar_kws={'label': '% of lightning strikes'})
plt.xlabel("Month")
plt.ylabel("Year")
plt.title("% of lightning strikes each Month (2016-2018)")
plt.tight_layout(pad=1.0)
plt.show()
```



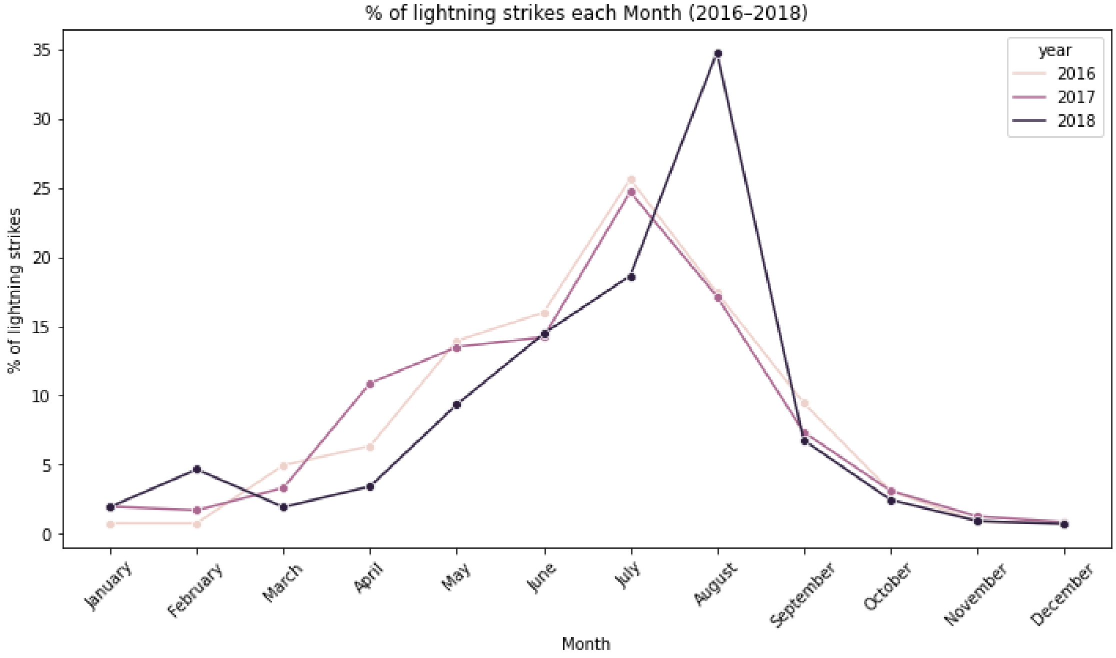
[35]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Ensure months are ordered correctly
month_order = ['January', 'February', 'March', 'April', 'May', 'June',
               'July', 'August', 'September', 'October', 'November', 'December']

# Convert month_text to categorical with correct order
percentage_lightning['month_text'] = pd.Categorical(
    percentage_lightning['month_text'], categories=month_order, ordered=True
)

# Sort data for consistent plotting
percentage_lightning = percentage_lightning.sort_values(['year', 'month_text'])

# Create numeric x-axis positions for months
month_to_num = {month: i for i, month in enumerate(month_order)}
percentage_lightning['month_num'] = percentage_lightning['month_text'].
 ↪map(month_to_num)
```

```python
# Plot bubble chart
plt.figure(figsize=(10, 6))
scatter = plt.scatter(
    x=percentage_lightning['month_num'],
    y=percentage_lightning['year'],
    s=percentage_lightning['percentage_lightning_per_month'] * 20,  # scale bubble size
    c=percentage_lightning['percentage_lightning_per_month'],
    cmap='YlGnBu',
    alpha=0.6,
    edgecolors='w',
    linewidth=0.5
)

# Customize axes
plt.xticks(ticks=range(12), labels=month_order, rotation=45)
plt.xlabel("Month")
plt.ylabel("Year")
plt.title("% of lightning strikes each Month (2016-2018)")

# Add colorbar
cbar = plt.colorbar(scatter)
cbar.set_label('% of lightning strikes')

plt.tight_layout(pad=1.0)
plt.show()
```



[36]:

```python
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Ensure months are ordered correctly
month_order = ['January', 'February', 'March', 'April', 'May', 'June',
               'July', 'August', 'September', 'October', 'November', 'December']

# Convert month_text to categorical with correct order
percentage_lightning['month_text'] = pd.Categorical(
    percentage_lightning['month_text'], categories=month_order, ordered=True
)

# Sort data for consistent line plotting
percentage_lightning = percentage_lightning.sort_values(['year', 'month_text'])

# Plot line chart
plt.figure(figsize=(10, 6))
sns.lineplot(
    data=percentage_lightning,
    x='month_text',
    y='percentage_lightning_per_month',
    hue='year',
    marker='o'
)

# Customize axes and layout
plt.xlabel("Month")
plt.ylabel("% of lightning strikes")
plt.title("% of lightning strikes each Month (2016-2018)")
plt.xticks(rotation=45)
plt.tight_layout(pad=1.0)
plt.show()
```



[ ]:

```
```