Krista Knuckey
Week 9 & 10

Audience:

The audience for this dataset/ infographic are TSA agents. The data provided gave quite a bit of insight on the day to day frustrations of everyday clients, which can be used to influence the operation and attitudes of TSA employees.

Purpose:

The purpose of the poster is to create a campaign to educate TSA agents on what the main issues and complaints are for our customers. Through this education, we can influence the agents on how to help improve our customer experience, even through something as simple as a smile can go a long way!

Medium/ Design:

The medium for this dataset is a poster that can be hung in break rooms, offices, and any behind- the- scenes area for TSA agents. This can also be downloaded to send in email and put in presentations for team meetings. The design incorporated grays and blues to demonstrate a calm discussion. Also, incorporating the word cloud visual that was created in my Python code allows the agent to see a majority of the issues and feel involved in the process of improving customer experience. Then, on the right hand side has bullet points that TSA agents can use immediately and help create discussion points for improving the department.

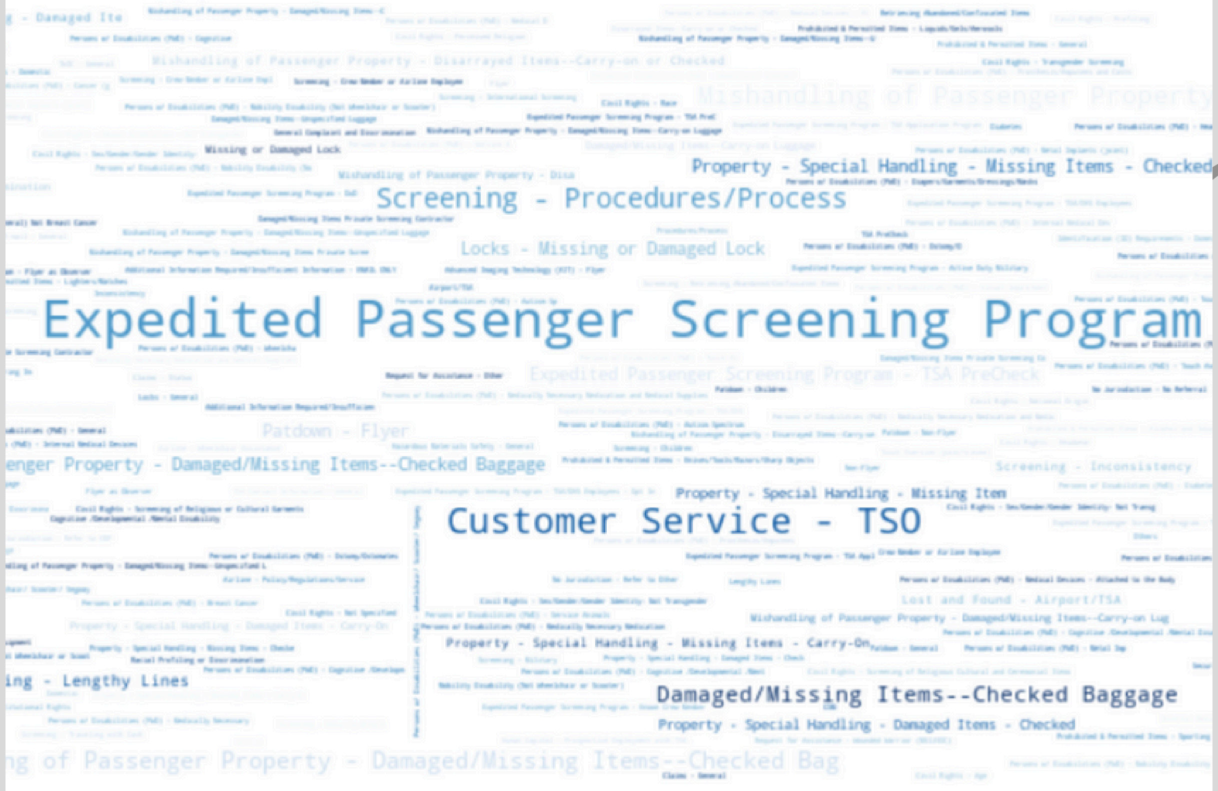
Ethical Considerations

The 3 datasets needed quite a bit of cleaning in order to use with the visual components, which is an ethical concern. The 3 datasets were merged, renamed columns, converted month data, and removed missing data. Removing missing data will always be an ethical concern because it can skew the data, however, with the visuals being focused on the categorical data of the complaint, it didn’t seem to make a difference. Also, the collection points of this data is an ethical concern as TSA is a strict program for safety measures, so making sure that the data collected will not impact safety measures is a big concern.

Customer Experience Data

Main customer issues to work on: Expedited Passnenger Screening Program

Our Top Priorities


Mishandling of Passenger Property
Property - Special Handling - Missing Items - Checked
Screening - Procedures/Process
Locks - Missing or Damaged Lock
Expedited Passenger Screening Program
Expedited Passenger Screening Program - TSA PreCheck
Patdown - Flyer
enger Property - Damaged/Missing Items--Checked Baggage
Screening - Inconsistency
Property - Special Handling - Missing Item
Customer Service - TSO
Lost and Found - Airport/TSA
Property - Special Handling - Missing Items - Carry-On
ing - Lengthy Lines
Damaged/Missing Items--Checked Baggage
Property - Special Handling - Damaged Items - Checked
ng of Passenger Property - Damaged/Missing Items--Checked Bag

Helpful Tips

- Smile - it can make all the difference!
- Monitor queue wait times and move customers to shorter lines when available
- Strike up a conversation! Ask about travel plans
- Educate customers on the importance of screening for their own safety
- Use kind words and tones - everyone deserves respect

DSC640_KristaKnuckey_Week9&10FV

November 3, 2024

[14]:

```
#importing libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:

```
#cleaning and combining dataset for better visuals and interpretation

complaints_by_airport = pd.read_csv('complaints-by-airport.csv')
complaints_by_category = pd.read_csv('complaints-by-category.csv')
complaints_by_subcategory = pd.read_csv('complaints-by-subcategory.csv')
iata_icao = pd.read_csv('iata-icao.csv')

merged_data = pd.merge(complaints_by_airport, iata_icao, how='left',␣
 ↪left_on='airport', right_on='iata')
merged_category_data = pd.merge(complaints_by_category, iata_icao, how='left',␣
 ↪left_on='airport', right_on='iata')
merged_subcategory_data = pd.merge(complaints_by_subcategory, iata_icao,␣
 ↪how='left', left_on='airport', right_on='iata')

# Rename columns
merged_data = merged_data.rename(columns={
    'airport_x': 'airport_code',
    'airport_y': 'airport_name',
    'year_month': 'complaint_month',
    'count': 'complaint_count'
})

merged_category_data = merged_category_data.rename(columns={
    'airport_x': 'airport_code',
    'airport_y': 'airport_name',
    'year_month': 'complaint_month',
    'count': 'complaint_count'
})

merged_subcategory_data = merged_subcategory_data.rename(columns={
    'airport_x': 'airport_code',
    'airport_y': 'airport_name',
    'year_month': 'complaint_month',
    'count': 'complaint_count'
})

# Convert 'complaint_month' to date
merged_data['complaint_month'] = pd.to_datetime(merged_data['complaint_month'],␣
  ↪format='%Y-%m')
merged_category_data['complaint_month'] = pd.
  ↪to_datetime(merged_category_data['complaint_month'], format='%Y-%m')
merged_subcategory_data['complaint_month'] = pd.
  ↪to_datetime(merged_subcategory_data['complaint_month'], format='%Y-%m')

# Check for missing values
missing_values_airport = merged_data.isnull().sum()
missing_values_category = merged_category_data.isnull().sum()
missing_values_subcategory = merged_subcategory_data.isnull().sum()

# Display the missing values
print("Missing values in airport data:\n", missing_values_airport)
print("Missing values in category data:\n", missing_values_category)
print("Missing values in subcategory data:\n", missing_values_subcategory)
```

```
Missing values in airport data:
 pdf_report_date      0
airport_code       109
complaint_month      0
complaint_count      0
country_code       396
region_name        396
iata               396
icao               435
airport_name       396
latitude           396
longitude          396
dtype: int64
Missing values in category data:
 pdf_report_date        0
airport_code        4042
category               0
complaint_month        0
complaint_count        0
clean_cat              0
clean_cat_status       0
country_code        5045
region_name         5045
iata                5045
icao                5132
```

```
airport_name        5045
latitude            5045
longitude           5045
dtype: int64
Missing values in subcategory data:
 pdf_report_date                  0
airport_code                 13464
category                         0
subcategory                      0
complaint_month                  0
complaint_count                  0
clean_cat                        0
clean_subcat                     0
clean_cat_status                 0
clean_subcat_status              0
is_category_prefix_removed       0
country_code                 14770
region_name                  14770
iata                         14770
icao                         14866
airport_name                 14770
latitude                     14770
longitude                    14770
dtype: int64
```

[4]:

```
#drop data that is not needed

cleaned_merged_data = merged_data.dropna(subset=['airport_code', 'latitude',␣
 ↪'longitude'])
cleaned_merged_category_data = merged_category_data.
 ↪dropna(subset=['airport_code', 'latitude', 'longitude'])
cleaned_merged_subcategory_data = merged_subcategory_data.
 ↪dropna(subset=['airport_code', 'latitude', 'longitude'])

missing_values_cleaned_airport = cleaned_merged_data[['airport_code',␣
 ↪'latitude', 'longitude']].isnull().sum()
missing_values_cleaned_category = cleaned_merged_category_data[['airport_code',␣
 ↪'latitude', 'longitude']].isnull().sum()
missing_values_cleaned_subcategory =␣
 ↪cleaned_merged_subcategory_data[['airport_code', 'latitude', 'longitude']].
 ↪isnull().sum()
```

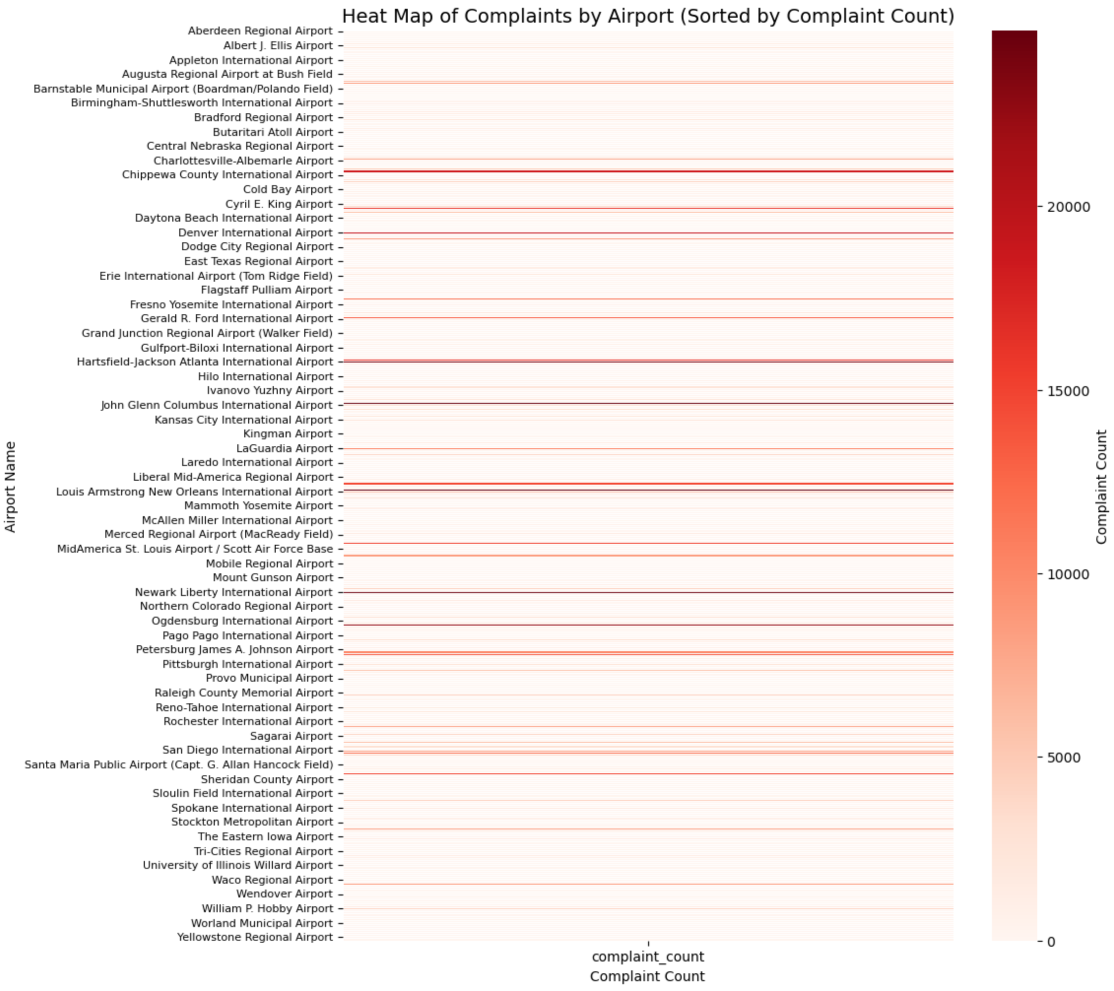
[6]:

```
#Visual 1 Heatmap- Airports by Number of complaints
```

[5]:

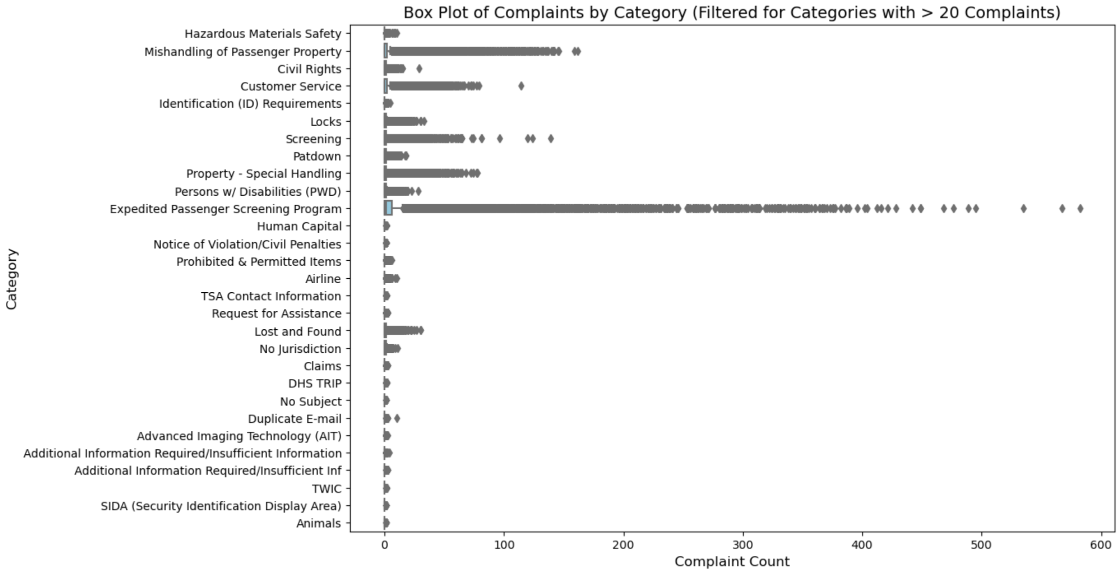
```
heatmap_data = cleaned_merged_data.groupby('airport_name')['complaint_count'].
 ↪sum().reset_index()
```

```
heatmap_data_sorted = heatmap_data.sort_values(by='complaint_count',␣
  ↪ascending=False)

plt.figure(figsize=(10, 12))
sns.heatmap(
    heatmap_data_sorted.pivot_table(index='airport_name',␣
  ↪values='complaint_count'),
    fmt="d",
    cmap="Reds",
    linewidths=0.5,
    cbar_kws={'label': 'Complaint Count'},
    annot=False
)

plt.title('Heat Map of Complaints by Airport (Sorted by Complaint Count)',␣
  ↪fontsize=14)
plt.xlabel('Complaint Count')
plt.ylabel('Airport Name')
plt.xticks(fontsize=10)
plt.yticks(fontsize=8, rotation=0)
plt.show()
```



```
[22]: #Visual 2- Box plot of complaints by category- filtered for 20 or more for␣
      ↪easier visual
```

```
[7]: category_counts = cleaned_merged_category_data.
      ↪groupby('category')['complaint_count'].sum()
     filtered_categories = category_counts[category_counts > 20].index
     filtered_data =␣
      ↪cleaned_merged_category_data[cleaned_merged_category_data['category'].
      ↪isin(filtered_categories)]

     plt.figure(figsize=(12, 8))
     sns.boxplot(
         data=filtered_data,
         x='complaint_count',
         y='category',
         color="skyblue"
)

plt.title('Box Plot of Complaints by Category (Filtered for Categories with >␣
 ↪20 Complaints)', fontsize=14)
plt.xlabel('Complaint Count', fontsize=12)
plt.ylabel('Category', fontsize=12)
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)
plt.show()
```

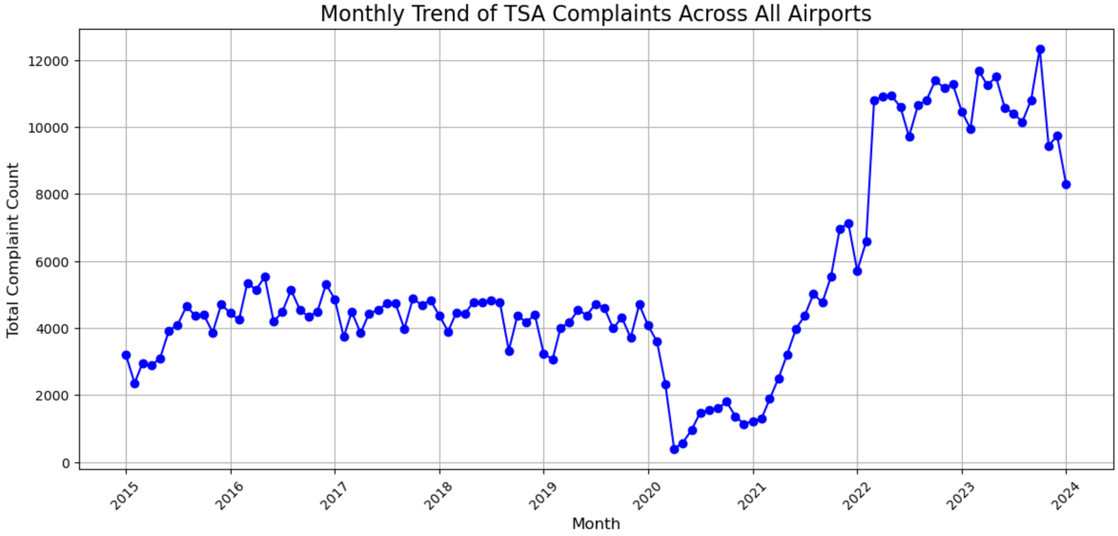


[8]:

```
#Visual 3- time series of trend of compalints by year- combined data from␣
 ↪months to year for easier visual
```

[9]:

```
import matplotlib.pyplot as plt

time_series_data = cleaned_merged_data.
 ↪groupby('complaint_month')['complaint_count'].sum().reset_index()

plt.figure(figsize=(14, 6))
plt.plot(time_series_data['complaint_month'],␣
 ↪time_series_data['complaint_count'], color='b', marker='o', linestyle='-')
plt.title('Monthly Trend of TSA Complaints Across All Airports', fontsize=16)
plt.xlabel('Month', fontsize=12)
plt.ylabel('Total Complaint Count', fontsize=12)
plt.xticks(rotation=45)
plt.grid(visible=True)
plt.show()
```

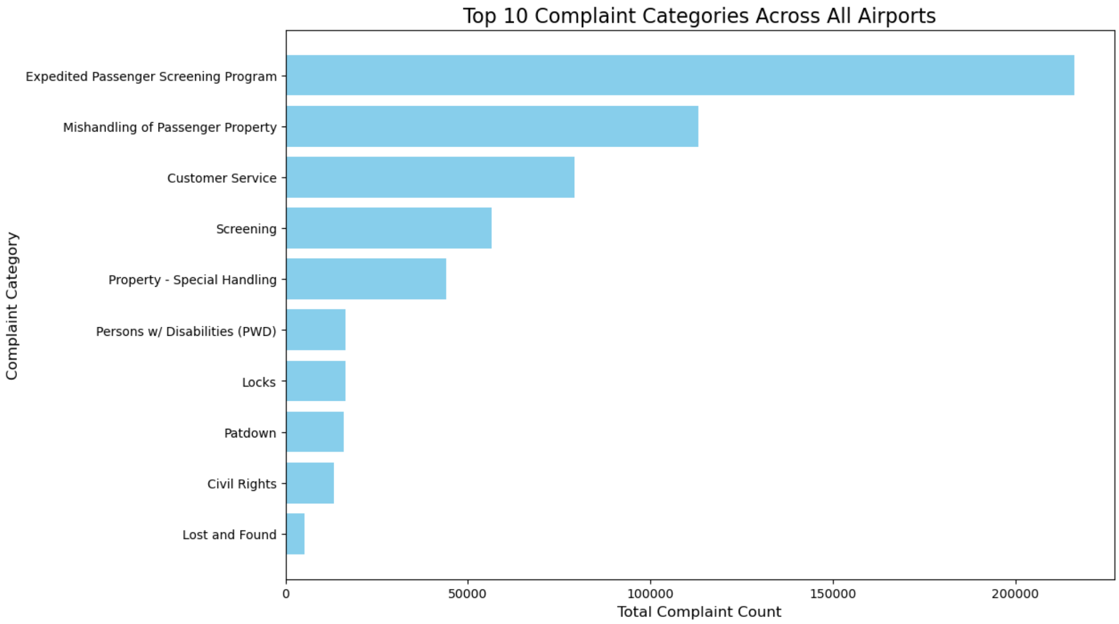


[25]:

```
#Visual 4- horizontal bar chart of top compalaints for all airports
```

[10]:

```
category_data = cleaned_merged_category_data.
 ↪groupby('category')['complaint_count'].sum().reset_index()

top_categories = category_data.sort_values(by='complaint_count',␣
 ↪ascending=False).head(10)

plt.figure(figsize=(12, 8))
plt.barh(top_categories['category'], top_categories['complaint_count'],␣
 ↪color='skyblue')
plt.title('Top 10 Complaint Categories Across All Airports', fontsize=16)
plt.xlabel('Total Complaint Count', fontsize=12)
plt.ylabel('Complaint Category', fontsize=12)
plt.gca().invert_yaxis()

plt.show()
```



[27]:

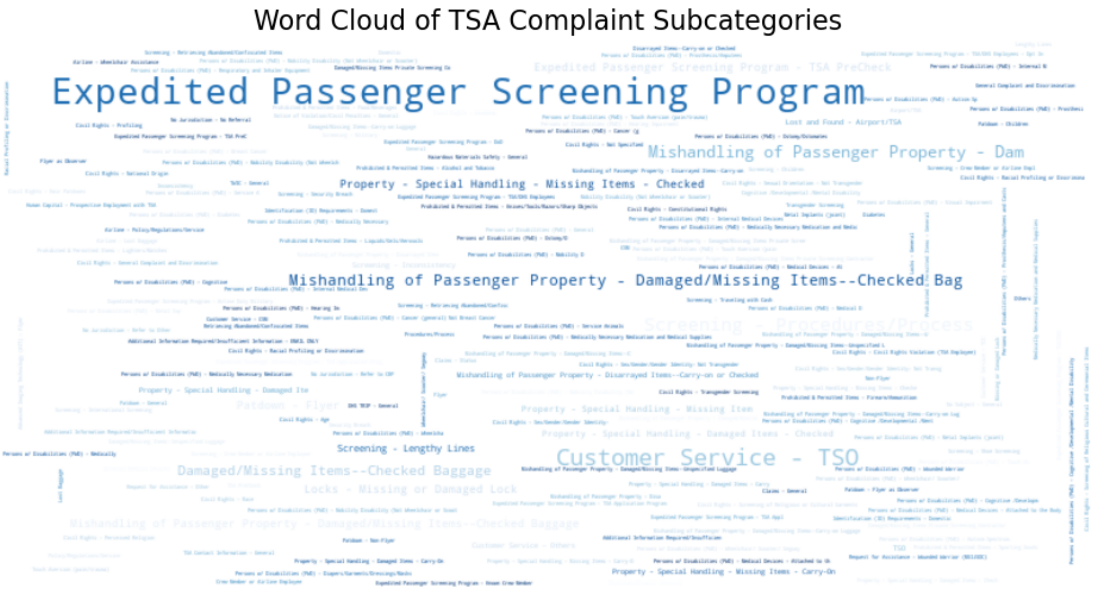
```
#Visual 5- wordcloud of the most used words from the complaints customers give
```

[11]:

```
!pip install wordcloud
from wordcloud import WordCloud
subcategory_data = cleaned_merged_subcategory_data.
  ↪groupby('subcategory')['complaint_count'].sum().reset_index()

subcategory_dict = dict(zip(subcategory_data['subcategory'],␣
  ↪subcategory_data['complaint_count']))

wordcloud = WordCloud(width=800, height=400, background_color='white',␣
  ↪colormap='Blues').generate_from_frequencies(subcategory_dict)

plt.figure(figsize=(12, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of TSA Complaint Subcategories', fontsize=16)
plt.show()
```

```
Requirement already satisfied: wordcloud in /opt/anaconda3/lib/python3.11/site-
packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in
/opt/anaconda3/lib/python3.11/site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in /opt/anaconda3/lib/python3.11/site-
packages (from wordcloud) (10.2.0)
Requirement already satisfied: matplotlib in /opt/anaconda3/lib/python3.11/site-
packages (from wordcloud) (3.8.0)
```

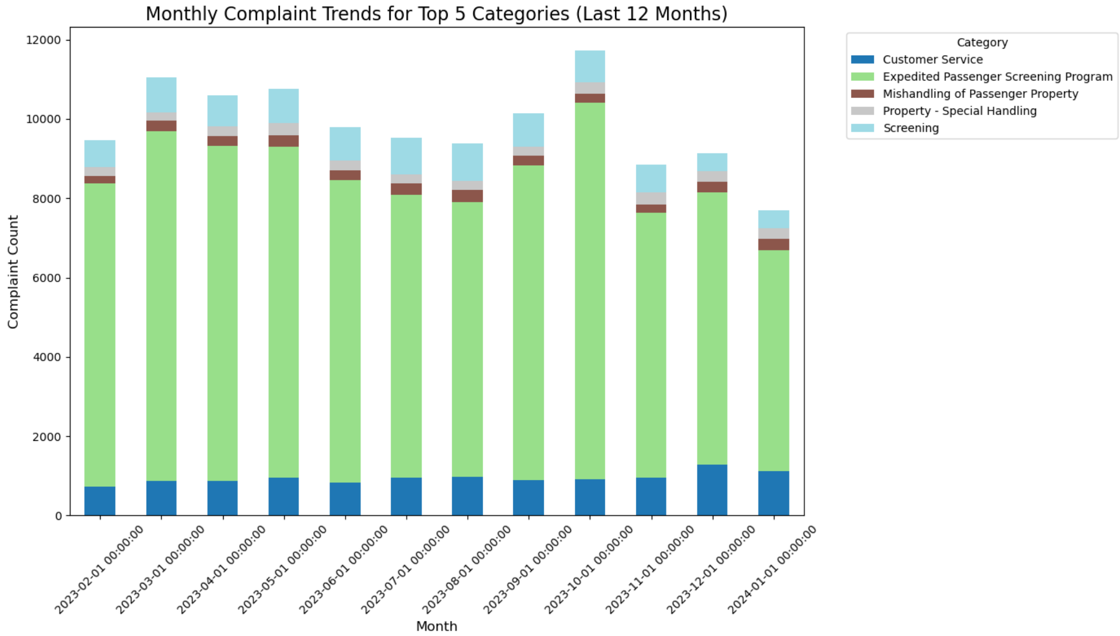
```
Requirement already satisfied: contourpy>=1.0.1 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud)
(0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud)
(4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud) (23.1)
Requirement already satisfied: pyparsing>=2.3.1 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
/opt/anaconda3/lib/python3.11/site-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in /opt/anaconda3/lib/python3.11/site-
packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
```



```
[12]: #Visual 6- stacked bar chart of monthly complaints- using the last 12 months␣
      ↪and top 5 complaints for best visual and data interpretation
```

```
[13]: top_5_categories = category_data.sort_values(by='complaint_count',␣
      ↪ascending=False).head(5)['category']
      top_category_data =␣
      ↪cleaned_merged_category_data[cleaned_merged_category_data['category'].
      ↪isin(top_5_categories)]
```

```
most_recent_12_months = top_category_data['complaint_month'].sort_values().unique()[-12:]
filtered_time_data = top_category_data[top_category_data['complaint_month'].isin(most_recent_12_months)]

time_category_data = filtered_time_data.groupby(['complaint_month', 'category'])['complaint_count'].sum().unstack()

time_category_data.plot(kind='bar', stacked=True, figsize=(14, 8), colormap='tab20')
plt.title('Monthly Complaint Trends for Top 5 Categories (Last 12 Months)', fontsize=16)
plt.xlabel('Month', fontsize=12)
plt.ylabel('Complaint Count', fontsize=12)
plt.xticks(rotation=45)
plt.legend(title='Category', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



[ ]: