FINAL PROJECT

Large_Passenger_Plane_Crashes_1933_to_2009

Professor: Mariana Shimabukuro

Group #6 Members:	**Student ID:**	**#CRN**
Wahid Popal Ali Ahmad Popal	100780969	45251
Fardin Alam	100780969	45251

Document outline

1. Introduction:

Traveling in planes can be very dangerous especially on unknow/not very famous airlines. Considering how important humans' lives are, we decided to choose a dataset that would not only inform us on how good or bad something is, but it also shows us what is the best and most save ways to use it.
The Large_Passenger_Plane_Crashes_1933_to_2009 dataset is not only showing us how many people died or survived, but it also shows us which airlines are the safest and which airlines we should not travel in and much more....

The questions we have chosen for our dataset are:

1. **What is the average survival rate of all the planes?**
2. **What are the overall boarded, and fatalities passengers in airplanes?**
3. **What are the maximum deaths on the ground caused by airplanes?**
4. **Which plane has the most crashes and how many people died?**
5. **Plane with the least crashes and its survival Rate?**
6. **Which year had the most crashes and how many people died?**

Team Members:

Wahid Popal Ali Ahmad Popal: 1/4/6 Document Outline, except introduction the entire Blog Post Report. Finally, done 1/2/3/6/7 Data Analysis questions with the 'Understanding your data' code.

Fardin Alam: 2/3/5 plus Reflection document outline, introduction for blog spot, data analysis questions 4 and 5.

2. Description of Data:

Our dataset contains the records of plane crashes from 1933-2009, we accessed this dataset from kaggle and it was derived from another dataset called Airplane*Crashes*and*Fatalities*Since_1908.csv made by Sauro Grandi, the dataset we worked on was created by Juan C Ventosa.

Link: https://www.kaggle.com/juancarlosventosa/large-passenger-plane-crashes-19332009

3. Analysis of the data:

Our dataset contains 16 columns: Date, Type, ClustID, Survival rate, Time, location, Operator, flight number, route, cn.in, passengers aboard, fatalities, what ground they crashed on (ex-ground 0, 1.etc), the number of survivors and a summarized reason for why the crash happened. We cleaned the data column using the to_datetime function in pandas to convert it into datetime type values. Our dataset has recorded every crash from the year 1933 - 2009, which is 76 years' worth of data, and for each record we have 16 columns of information which leads us to believe our dataset is complete and our data records are "full". All our records follow the same formatting.

4. Exploratory Data Analysis:

1) Begin to read dataset

Before working on anything it's very important that we have the necessary libraries and the correct way of opening a file.

```
#importing used libraries
import numpy as np
import pandas as pd
from functools import reduce
import matplotlib.pyplot as plt
import seaborn as sns

# loading the data using pandas
df = pd.read_csv('Large_Passenger_Plane_Crashes_1933_to_2009.csv')

# preview of the data
df.head()
```

	Date	Time	Location	Operator	Flight..	Route	Type	Registration	cn.In	Aboard	Fatalities	Ground	Survivors	SurvivalRate	Summary
0	4/4/33	12:30	Off Barnegat, New Jersey	Military - U.S. Navy	NaN	NaN	Goodyear-Zeppelin U.S.S. Akron (airship)	ZRS-4	NaN	76	73	0	3	0.039474	While cruising at 1,600 feet off New Jersey, s...
1	3/12/50	14:50	Llandow Airport, Cardiff, Wales	Fairflight Ltd.	NaN	Llandow - Dublin	Avro 689 Tudor 5	G-AKBY	1417	83	80	0	3	0.036145	During the approach to Runway 28 at Llandow Ai...
2	3/26/52	NaN	Moscow, Russia	Aeroflot	NaN	NaN	NaN	NaN	NaN	70	70	0	0	0.000000	The plane overshot the runway and collided wit...
3	12/20/52	6:30	Moses Lake, Washington	Military - U.S. Air Force	NaN	NaN	Douglas C-124A Globemaster	50-100	43238	115	87	0	28	0.243478	Within two minutes after takeoff the aircraft ...
4	6/18/53	16:34	Tachikawa AFB, Tokyo, Japan	Military - U.S. Air Force	NaN	Tachikawa AB - Kimpo AB	Douglas C-124A Globemaster II	51-137A	43471	129	129	0	0	0.000000	Crashed shortly after taking off from Tachikaw...

2) Understanding the dataset

We ran many lines of code just to understand the data such as the length of the dataset, info(), find the mean, min, max, 25%, 50%, 75%, and many other small values that was necessary for properly understanding the dataset.

```
# getting more information about the dataset's datatype
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 456 entries, 0 to 455
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          456 non-null    object
 1   Time          390 non-null    object
 2   Location      456 non-null    object
 3   Operator      456 non-null    object
 4   Flight..      301 non-null    object
 5   Route         417 non-null    object
 6   Type          455 non-null    object
 7   Registration  443 non-null    object
 8   cn.In         433 non-null    object
 9   Aboard        456 non-null    int64
 10  Fatalities    456 non-null    int64
 11  Ground        456 non-null    int64
 12  Survivors     456 non-null    int64
 13  SurvivalRate  456 non-null    float64
 14  Summary       456 non-null    object
 15  ClustID       456 non-null    object
dtypes: float64(1), int64(4), object(11)
memory usage: 57.1+ KB
```

```
# The length of dataset
len(df)
```

```
456
```

```
# the different columns of dataset
df.keys()
```

```
Index(['Date', 'Time', 'Location', 'Operator', 'Flight..', 'Route', 'Type',
```

```
# observing the statistical description of the dataset
df.describe()
```

	Aboard	Fatalities	Ground	Survivors	SurvivalRate
count	456.000000	456.000000	456.000000	456.000000	456.000000
mean	137.256579	92.366228	6.947368	44.890351	0.266674
std	71.006407	70.641796	128.868778	78.568882	0.389414
min	69.000000	0.000000	0.000000	0.000000	0.000000
25%	90.000000	59.000000	0.000000	0.000000	0.000000
50%	116.000000	85.500000	0.000000	0.000000	0.000000
75%	157.000000	120.250000	0.000000	73.500000	0.526644
max	644.000000	583.000000	2750.000000	516.000000	1.000000

3) Cleaning process

Our main goal from the dataset Large_Passenger_Plane_Crashes_1933_to_2009 was to figure out which airplane is the safest one to travel in. So, we started cleaning the column "Type" which shows the names of different airplanes. Secondly, we started to clean the "Date" column as it's also very important for our data analysis questions.

```
df[df['Type'].notnull()]
```

	Date	Time	Location	Operator	Flight..	Route	Type	Registration	cn.In	Aboard	Fatalities	Ground	Survivors	SurvivalRate
0	4/4/33	12:30	Off Barnegat, New Jersey	Military - U.S. Navy	NaN	NaN	Goodyear-Zeppelin U.S.S. Akron (airship)	ZRS-4	NaN	76	73	0	3	0.039474
1	3/12/50	14:50	Llandow Airport, Cardiff, Wales	Fairflight Ltd.	NaN	Llandow - Dublin	Avro 689 Tudor 5	G-AKBY	1417	83	80	0	3	0.036145
3	12/20/52	6:30	Moses Lake, Washington	Military - U.S. Air Force	NaN	NaN	Douglas C-124A Globemaster	50-100	43238	115	87	0	28	0.243478
4	6/18/53	16:34	Tachikawa AFB, Tokyo, Japan	Military - U.S. Air Force	NaN	Tachikawa AB - Kimpo AB	Douglas C-124A Globemaster II	51-137A	43471	129	129	0	0	0.000000

```
#cleaning all the rows that doesn't have Date
df[df['Date'].notnull()]
```

	Date	Time	Location	Operator	Flight..	Route	Type	Registration	cn.In	Aboard	Fatalities	Ground	Survivors	SurvivalRate	!
0	33-04-04	12:30	Off Barnegat, New Jersey	Military - U.S. Navy	NaN	NaN	Goodyear-Zeppelin U.S.S. Akron (airship)	ZRS-4	NaN	76	73	0	3	0.039474	(J
1	50-03-12	14:50	Llandow Airport, Cardiff, Wales	Fairflight Ltd.	NaN	Llandow - Dublin	Avro 689 Tudor 5	G-AKBY	1417	83	80	0	3	0.036145	[ap R a
2	52-03-26	NaN	Moscow, Russia	Aeroflot	NaN	NaN	NaN	NaN	NaN	70	70	0	0	0.000000	ov ru coll
3	52-12-20	6:30	Moses Lake, Washington	Military - U.S. Air Force	NaN	NaN	Douglas C-124A Globemaster	50-100	43238	115	87	0	28	0.243478	\ af tl
	53-		Tachikawa	Military -		Tachikawa	Douglas C-								sh

4) Analyzing the dataset Large_Passenger_Plane_Crashes_1933_to_2009 and working on the questions

It's important to know how many people makes it out of airplane crashes alive so we add all the "survivalRate" column and divided by the rows of the dataset and found the average of the survivals on planes crashes.

```
survival_Average = df['SurvivalRate'].sum()
print('The average survival rate: ', survival_Average/len(df), '%')
```

```
The average survival rate:  0.2666737569298246 %
```

We also, wanted to know how many people were aborded and how man people died which is very important to know how dangerous is trailing in airplanes.

```
aboard_passangers = df['Aboard'].sum()
fatalitied_passangers = df["Fatalities"].sum()

print("The Overall aboarded passangers:   ", aboard_passangers)
print("The Overall fatalitied passangers: ", fatalitied_passangers)
```

```
The Overall aboarded passangers:    62589
The Overall fatalitied passangers:  42119
```

As you can see more the half of aborded people did not make it through the crash.

Not only the death of boarded people, but we also wanted to know while the plane was crashing on the ground how many people died, and which plane has the max death on ground. We found the sum of "Ground" column and also used the .max() function to find the plane with most destructions on ground.

```
df[df['Ground'] == df['Ground'].max()]
```

	Date	Time	Location	Operator	Flight..	Route	Type	Registration	cn.In	Aboard	Fatalities	Ground	Survivors	SurvivalRate	Summary	ClustID
402	2011-01-09	8:47	New York City, New York	American Airlines	11	Boston - Los Angeles	Boeing 767-223ER	N334AA	22332/169	92	92	2750	0	0.0	The aircraft was hijacked shortly after it lef...	High Fatalit

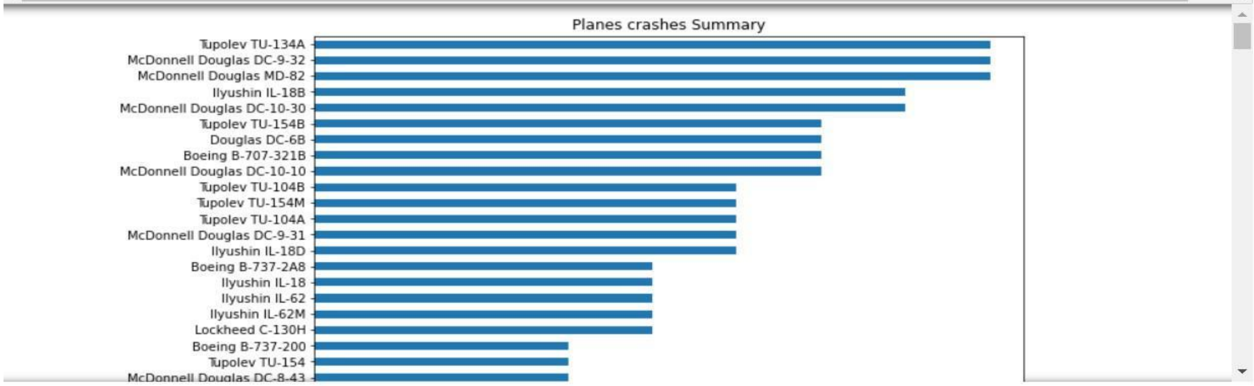
```
# how many people died in total on the ground
onGround_Deaths = df["Ground"].sum()
print('On Ground deaths cased by airplanes crashes: ', onGround_Deaths)
```

```
On Ground deaths cased by airplanes crashes:  3168
```

We have also, plotted a graph to see the most crashed planes and the least crashed planes and what are how many deaths caused by them. This way we can find the safest airline and can recommend which airline to travel with. This was one of the challenging parts of the assignment as there was no one plane to find so we draw a graph to find all the possible outcomes. We were stack there for days and even had the idea of changing the dataset. All the code will be below and the comment on code clearly explains what is going on.

```
#Calculating the crashes for each plane by ascending order and save it in plain_count
plain_count = df['Type'].value_counts().sort_values(ascending = True)
```

```
#ploting the diferent planes distribution using .plot.barh() method and add size, title, and  x and y label to our c
plain_count.plot.barh(title = 'Planes crashes Summary', figsize = (10,80)).set(xlabel = 'Crashes', ylabel = 'Planes'
plt.show()
```




```
#From the graph we can see that there are three planes that crashed max 8 times such as Tupolev TU-134A,
# McDonnell Douglas DC-9-32, and McDonnell Douglas MD-82.

print('Tupolev TU-134A')
df1 = df[df.apply(lambda row: row.astype(str).str.contains('Tupolev TU-134A').any(), axis=1)][['Type', 'Fatalities']
df1.sum()
```

```
Tupolev TU-134A

Type          Tupolev TU-134ATupolev TU-134A / Tupolev Tu-13...
Fatalities                                                  834
dtype: object
```

```
print('McDonnell Douglas DC-9-32')
df2 = df[df.apply(lambda row: row.astype(str).str.contains('McDonnell Douglas DC-9-32').any(), axis=1)][['Type', 'Fa
df2.sum()
```

```
McDonnell Douglas DC-9-32

Type          McDonnell Douglas DC-9-32McDonnell Douglas DC-...
Fatalities                                                  659
dtype: object
```

```
print('McDonnell Douglas MD-82')
df3 = df[df.apply(lambda row: row.astype(str).str.contains('McDonnell Douglas MD-82').any(), axis=1)][['Type', 'Fata
df3.sum()
```

```
McDonnell Douglas MD-82

Type          McDonnell Douglas MD-82McDonnell Douglas MD-82...
Fatalities                                                  693
dtype: object
```

Since the three planes has the most crashes, overall the fatalities of those three planes are: **2186**
We believe that those three planes are super dangerous and should not travel in it.

```
#First I want to count the number of occurences of each type of plane
df['Type'].value_counts()
```

```
Tupolev TU-134A                          8
McDonnell Douglas DC-9-32                8
McDonnell Douglas MD-82                  8
Ilyushin IL-18B                          7
McDonnell Douglas DC-10-30               7
                                        ..
McDonnell Douglas DC-8-Super 63CF        1
Antonov An-12                            1
Tupolev TU-134A / Tupolev Tu-134A        1
Boeing B-707-340C                        1
Airbus A330-203                          1
Name: Type, Length: 308, dtype: int64
```

```
df4 = df[df.apply(lambda row: row.astype(str).str.contains('Airbus A330-203').any(), axis=1)][['Type', 'SurvivalRate
df4
```

	Type	SurvivalRate
455	Airbus A330-203	0.0

```
df5 = df[df.apply(lambda row: row.astype(str).str.contains('Boeing B-707-340C').any(), axis=1)][['Type', 'SurvivalRa
df5
```

	Type	SurvivalRate
194	Boeing B-707-340C	0.0

```
df6 = df[df.apply(lambda row: row.astype(str).str.contains('Tupolev TU-134A / Tupolev Tu-134A').any(), axis=1)][['Ty
df6
```

	Type	SurvivalRate
191	Tupolev TU-134A / Tupolev Tu-134A	0.0

```
df7 = df[df.apply(lambda row: row.astype(str).str.contains('Antonov An-12').any(), axis=1)][['Type', 'SurvivalRate']
df7
```

	Type	SurvivalRate
72	Antonov An-12 - Ilyshin IL-14	0.0
185	Antonov An-12	0.0

```
df8 = df[df.apply(lambda row: row.astype(str).str.contains('McDonnell Douglas DC-8-Super 63CF').any(), axis=1)][['Ty
df8
```

	Type	SurvivalRate
184	McDonnell Douglas DC-8-Super 63CF	0.301527

Since we checked all the five planes survival Rate and the average is: **0.0603054 %**
the possibility of those planes crashing is very small. So, we can say, those planes are save for traviling.

Last, but not least, we found which year has the most crashes of airplanes.
We first made new columns for year, month, and day then used value_counts() function to find the year. A long the way we found many other questions like the year with the most deaths, the month and time of the most crashes and many other questions. The code bellow

```
#converting date column into datetime type of data
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

#Creating a new column for Year and put all the years in there
df['Year'] = df['Date'].dt.strftime('%y')

#Creating a new column for Months and put all the Months in there
df['Month'] = df['Date'].dt.month_name().str[:3]

#Creating a new column for Days and put all the Days in there
df['Day'] = df['Date'].dt.day

df
```

erator	Flight..	Route	Type	Registration	cn.In	Aboard	Fatalities	Ground	Survivors	SurvivalRate	Summary	ClustID	Year	Month	Day
ilitary - . Navy	NaN	NaN	Goodyear-Zeppelin U.S.S. Akron (airship)	ZRS-4	NaN	76	73	0	3	0.039474	While cruising at 1,600 feet off New Jersey, s...	High Fatality	1933	Apr	4.0
ıirflight Ltd.	NaN	Llandow - Dublin	Avro 689 Tudor 5	G-AKBY	1417	83	80	0	3	0.036145	During the approach to Runway 28 at Llandow Ai...	High Fatality	1950	Mar	12.0
eroflot	NaN	NaN	NaN	NaN	NaN	70	70	0	0	0.000000	The plane overshot the runway and collided wit...	High Fatality	1952	Mar	26.0

has comments that would walk you the process of how and why things are done.

```
#Counting each rows for different years
df['Year'].value_counts()
```

```
1973    17
1972    16
1974    15
1982    14
1985    14
        ..
1902     1
1950     1
1926     1
1911     1
1933     1
Name: Year, Length: 74, dtype: int64
```

```
print('Year 1973')
df8 = df[df.apply(lambda row: row.astype(str).str.contains('73').any(), axis=1)][['Year', 'Fatalities']]
df8.sum()
```

```
Yes: 1973
```

```
/tmp/ipykernel_43/1136332198.py:3: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numer
ic_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before call
ing the reduction.
  df8.sum()
```

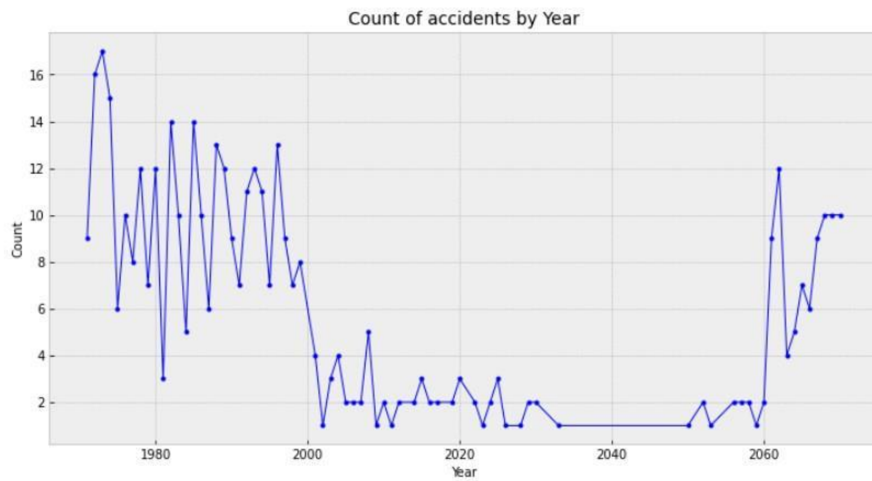
```
Fatalities    9170
dtype: int64
```

From the above data we can clearly see that **1973** has the most crashes.
Fatalities of crashes in the 1973 year are **9170** which is a lot.

1.The graph shows the years and it's count of accidents in this year.

```
Temp = df.groupby(df.Date.dt.year)[['Date']].count() #Temp is going to be temporary data frame
Temp = Temp.rename(columns={"Date": "Count"})

plt.figure(figsize=(12,6))
plt.style.use('bmh')
plt.plot(Temp.index, 'Count', data=Temp, color='blue', marker = ".", linewidth=1)
plt.xlabel('Year', fontsize=10)
plt.ylabel('Count', fontsize=10)
plt.title('Count of accidents by Year', loc='Center', fontsize=14)
plt.show()
```

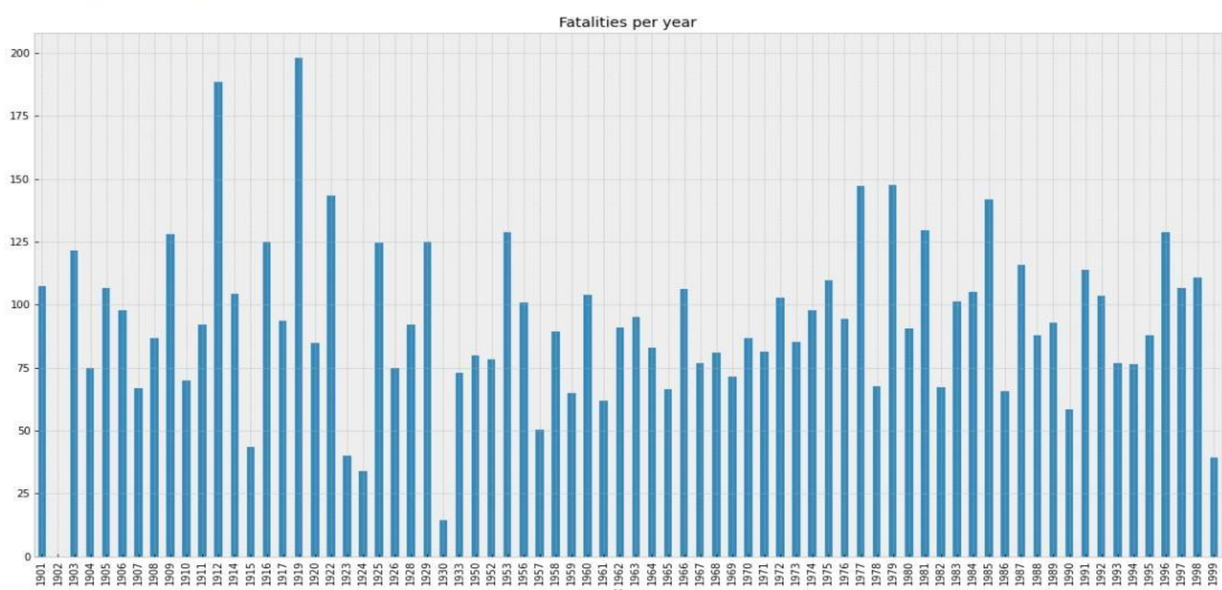


2.The plotted graph represents the year and how many people died in airplanes crashes in this year. The year starts with 1901 which supposed to be 1933

```
FatalitiesPeryear = df.groupby('Year')['Fatalities'].mean()
```

```
FatalitiesPeryear.plot(kind = 'bar', title = 'Fatalities per year', figsize = (20,10), x = "Year")
```

```
<AxesSubplot:title={'center':'Fatalities per year'}, xlabel='Year'>
```



5. Potential Data Science:

The potential I see in this dataset is using this information to document the evolution of the aviation industry over the years, and to examine further at the different types of planes and the reasons they crashed to model new types of aircrafts that are not susceptible to those errors. The dataset is also a "real-world" dataset, so the information obtained from the analysis can be useful to real world situations. The aviation industry is an industry that is always looking for ways to improve and adapt to the current technological situation of the world, whether it be for assisting the pilot with autopilot features or even being able to use machine learning to fly the airplane by itself. The data that we collected can be crucial for data scientists, since it shows where humans go wrong while flying large passenger aircrafts. With this data they can create better algorithms and software to take better and safer command of aircrafts. Aircrafts like the Boeing and Airbus which have been recorded in our dataset above, have already begun making prototypes of self-piloted passenger air crafts. We can use the records of our failures to build a future where mankind will be able to travel overseas with a peace of mind, we can make our failures be a part of our success.

6. Conclusion:

In my opinion the Large_Passenger_Plane_Crashes_1933_to_2009 dataset is one of the most important datasets of all type. Humans' lives are precious, and the world needs to know which planes are safe and which are not for trailing. As we previously mentioned this dataset can not only help you save your life, but it also gives the manufactures ideas of what they have done wrong and what can/should be changed to provide a safer and better journey for the world. The dataset might not be three or five thousand rows, but it's very old. It starts from 1933_to_2009 which you can learn a lot from it. The saddest thing about this dataset which I never thought of it is that more than half of a boarded people died in those accidents. Also, I was very surprised when I saw the amount of dead people that were killed by planes while landing which was almost 3168 and that was horrifying. Not to mention, we ran into many issues with this data while we were making questions and especially coding them. There were no one plane that has the most or the least crashes which lead to huge problems. We also had a very hard time changing the data to datetime date. Whenever we would change it always gave us 2033 instead of 1933 and had some issues, but we finally fix it. Overall, the dataset was very smooth, and we had a lot of fun coding it, and learning from it.

Reflection: We learned a lot from this dataset, being able to isolate certain rows in our dataframe to extract information, to filter NaN data and to create comprehensive graphs to help visualize our data for the reader. If we were to improve our project, we'd like to have brought in more recent crash data from 2010-2020, to really show the evolution of the aviation industry.