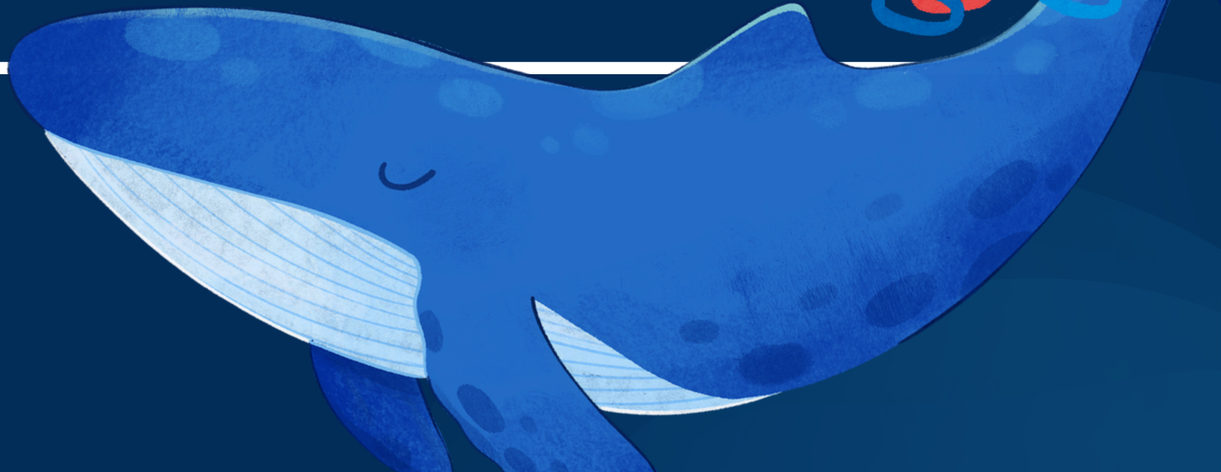
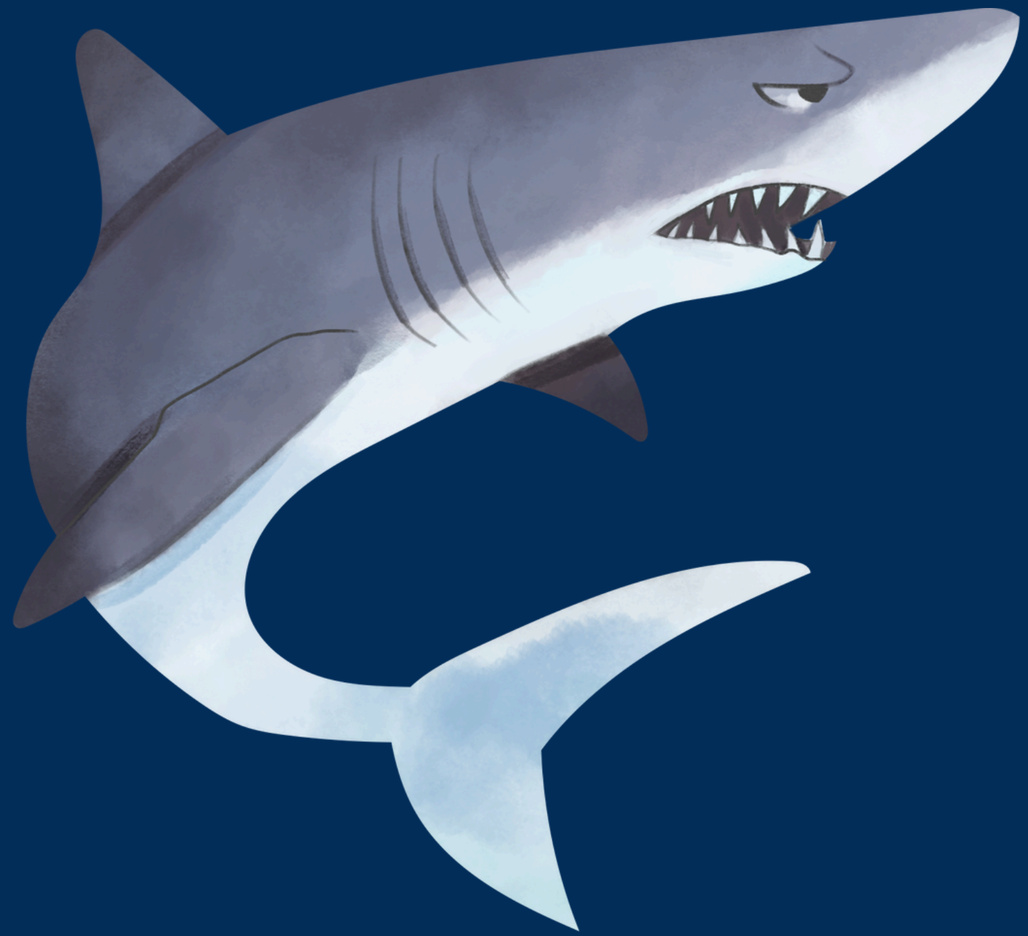
Reggae Sharks| *Laura Sánchez, Benjamín Mancera, Irene Gauna, Clara Gallego, Miqueas Molina*

SHARK ATTACKS INSURANCE (OLYMPIC GAMES EDITION)



Contents:



1. Overview

- The USA is about to organise the Water Olympic Games for 2025.
- Many athletes from all over the world are expected to attend this massive event.
- The organizing committee is concerned about estimating insurance costs based on several key factors.

Here's a breakdown of those factors:

Age	Sport activity	Location of the attack	Gender

Shark attacks dataset

The dataset from the Shark Attack File records global shark attack incidents.

It captures information about each reported shark attack, including details like:

- Date
- Location (country, state, specific area)
- Activity of the victim at the time of the attack
- Demographics
  - Name
  - Age
  - Sex
- Type of attack
- Resulting injury
- Species of shark involved

Hypothesis

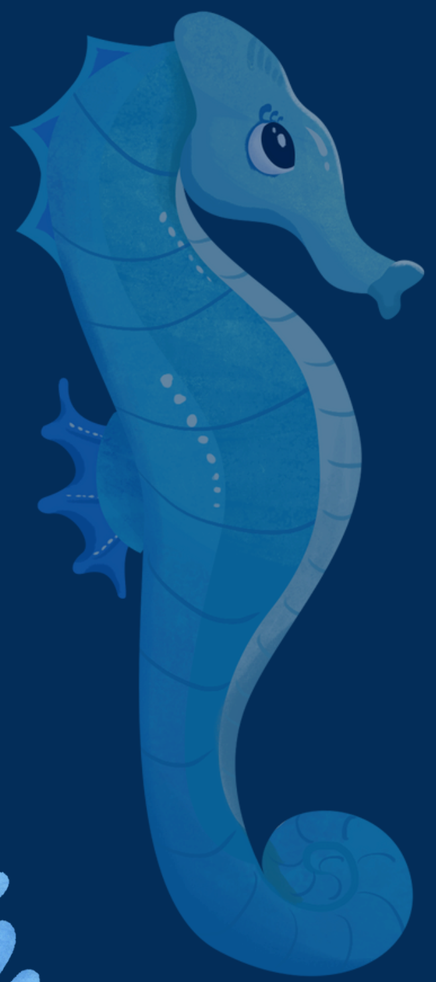
1. Young people suffer more shark attacks
2. Men are more prone to shark attacks
3. Coastal and warm states experience more shark attacks
4. Athletes who practice surfing are at a higher risk
5. Bonus: White shark is way more dangerous (fatal/dead)

2. Data Wrangling and Cleaning

When working with this dataset and analyzing the data it provides, we encountered the following difficulties:

- Columns with uncertain meaning and unnamed.
- Different ways of recording data, for example, in the columns sex ('F' 'M' nan 'lli' 'M x 2' 'N' '.'), species, activities and injury.
- Very old data with unreliable dates (5; 77; 1000).
- Unnecessary columns for our study like ‘source’.

```
['F' 'M' nan 'lli' 'M x 2' 'N' '.']
The value 'M x 2' occurs 1 times in the 'sex' column.
0       Female
1         Male
2       Female
3         Male
4         Male
         ...
6965      Male
6966      Male
6967      Male
6968      Male
6969      Male
Name: sex, Length: 6967, dtype: object
['Female' 'Male' 'Unknown']
```



To solve these problems:

- We unified the values in the columns.
- Removed columns that were either unused or had incongruent meanings.
- Filtered by date, eliminating the oldest data, retaining 85% of the data, from 1920 to 2024 and country USA.
- Grouped data by age, species, and activity.
- Changed undefined or unnecessary values.

```
# Count occurrences of each year in the 'year' column
year_counts = df['year'].value_counts().sort_index()

# Show all the years
pd.set_option('display.max_rows', None)

# Display the counts
print(year_counts)

# Remove rows where the 'year' is 2025 or 2026
df = df[~df['year'].isin([2025, 2026])]

# Count occurrences of each year again to verify removal
year_counts_updated = df['year'].value_counts().sort_index()

# Display the updated counts
# print(year_counts_updated)

# Remove rows where the 'year' is before 1920
df = df[df['year'] >= 1920]
```

Data cleaning techniques

Removing columns
df.drop()

```
[ ]  df.drop(columns=["pdf"], inplace = True)
     df.drop(columns=["href_formula"], inplace = True)
     df.drop(columns=["href"], inplace = True)
     df.drop(columns=["case_number"], inplace = True)
     df.drop(columns=["case_number.1"], inplace = True)
     df.drop(columns=["original_order"], inplace = True)
     df.drop(columns=["unnamed:_21"], inplace = True)
     df.drop(columns=["unnamed:_22"], inplace = True)
     df.drop(columns=["source"], inplace = True)
     df = df.rename(columns={'unnamed:_11': 'death'})
     df.head()
```

Renaming Columns
str.lower() + str.replace()

```
# Rename columns to lowercase and replace spaces with underscores
df.columns = df.columns.str.lower().str.replace(' ', '_')
```

Data Type Conversion
.astype()

```
     df['age'] = df['age'].astype(int)
     df['year'] = df['year'].astype(int)
     #df['month'] = df['month'].astype(int)
```

Cleaning and grouping variables

Sex
.map() + .fillna()

```
# Remove leading and trailing spaces from the 'sex' column
df['sex'] = df['sex'].str.strip()

# Get unique values from the 'sex' column
unique_values_sex = df['sex'].unique()
print(unique_values_sex)

# Count occurrences of the value 'M x 2' in the 'sex' column
count_m_x_2 = df['sex'].value_counts().get('M x 2', 0)
print(f"The value 'M x 2' occurs {count_m_x_2} times in the 'sex' column.")

# Changing values M to Male, F to Female and rest to Unknown
# Map values in the 'sex' column
mapping = {'F': 'Female', 'M': 'Male'}
df['sex'] = df['sex'].map(mapping).fillna('Unknown')
```

Age
grouping by ranges

```
df = df[df['age'].apply(lambda x: str(x).isdigit())]
# Create age ranges and labels
bins = [0, 17, 34, 54, 74, 100]  # Adjust the upper limit as necessary
labels = ['0-17', '18-34', '35-54', '55-74', '75+']

# Create a new column for age categories
df_usa['age_category'] = pd.cut(df_usa['age'], bins=bins, labels=labels)
```

Data cleaning techniques

Year filter

```
# Count occurrences of each year in the 'year' column
year_counts = df['year'].value_counts().sort_index()

# Show all the years
pd.set_option('display.max_rows', None)

# Display the counts
print(year_counts)

# Remove rows where the 'year' is 2025 or 2026
df = df[~df['year'].isin([2025, 2026])]

# Count occurrences of each year again to verify removal
year_counts_updated = df['year'].value_counts().sort_index()

# Display the updated counts
# print(year_counts_updated)

# Remove rows where the 'year' is before 1920
df = df[df['year'] >= 1920]
```

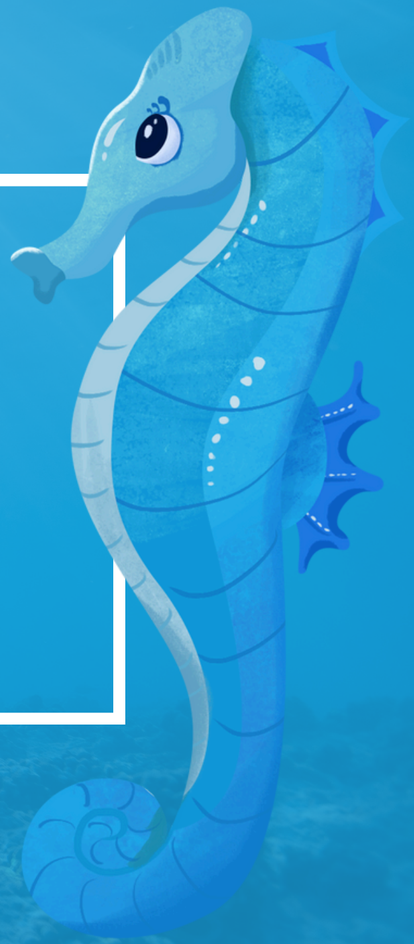
Activity (rename and group)

```
def clean_activity(activity):
    if pd.isna(activity):  # Check for NaN values
        return 'other activity'
    activity = activity.lower()  # Convert to lowercase for case-insensitive matching

    # Define the replacement rules
    if 'swimming' in activity or 'swim' in activity:
        return 'swimming'
    elif 'diving' in activity or 'dive' in activity:
        return 'diving'
    elif 'surfing' in activity or 'surf' in activity:
        return 'surfing'
    elif 'canoe' in activity:
        return 'canoe'
    elif 'kayak' in activity or 'kayaking' in activity:
        return 'kayaking'
    elif 'sail' in activity or 'sailing' in activity:
        return 'sailing'
    elif 'row' in activity or 'rowing' in activity:
        return 'rowing'
    elif 'kiteboard' in activity or 'kiteboarding' in activity or 'kite boarding' in activity:
        return 'kiteboarding'
    elif 'kite surf' in activity or 'kitesurf' in activity or 'kite surfing' in activity or 'kitesurfing' in activity:
        return 'kitesurfing'
    else:
        return 'other activity'

# Apply the cleaning function to the activity column
df['activity'] = df['activity'].apply(clean_activity)
```

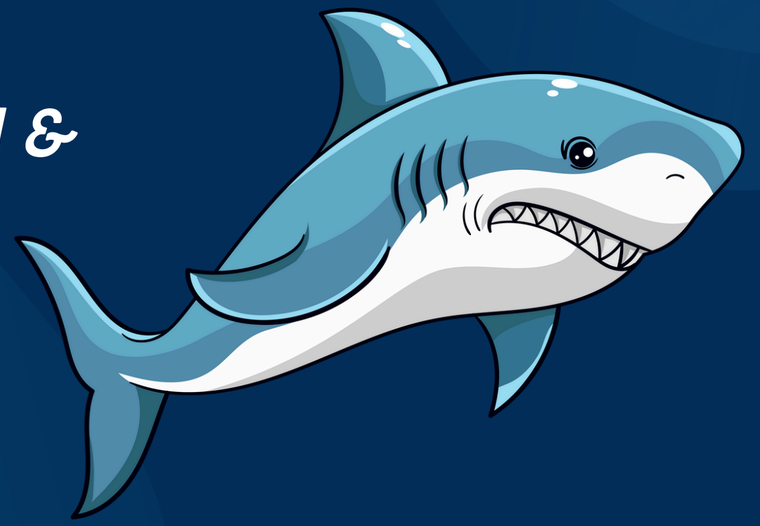
Handling Date and Time Data

```
def get_hour(x):
    if isinstance(x, str):
        return x[0:2]
    return None

df["hour"]=df["time"].apply(lambda x: get_hour(x))

def categorize_hour(hour):
    if isinstance(hour, str) and hour.isdigit():
        hour = int(hour)
    elif isinstance(hour, str):
        try:
            hour = int(hour.split("h")[0])
        except (ValueError, IndexError):
            return None

    if isinstance(hour, (int, float)):
        if 6 <= hour < 14:
            return "Morning"
        elif 14 <= hour < 19:
            return "Afternoon"
        elif 19 <= hour < 24:
            return "Evening"
        else:
            return "Night"
    elif hour in ['Mo', 'Af', 'Ni', 'La', 'Ev']:
        if hour == 'Mo':
            return "Morning"
        elif hour == 'Af':
            return "Afternoon"
        elif hour == 'Ni':
            return "Night"
        elif hour == 'La':
            return "Night"
        elif hour == 'Ev':
            return "Evening"
    return None
```

3. EXPLORATORY DATA ANALYSIS



Exploratory data analysis methods used & insights



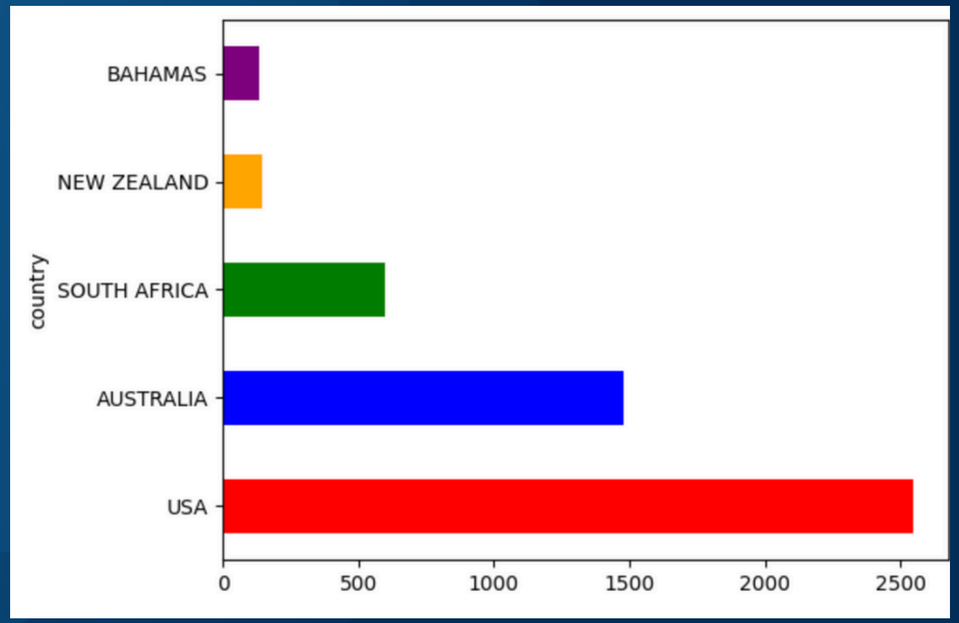
- **df.head()** to get a first impression of the data
- **df.isna().sum()**

```
df.isna().sum()
```

	0
Date	0
Year	2
Type	18
Country	50
State	482
Location	565
Activity	586
Name	220
Sex	579
Age	2995
Injury	35
Unnamed: 11	562
Time	3527
Species	3132

- **df.info()** to get types and other info

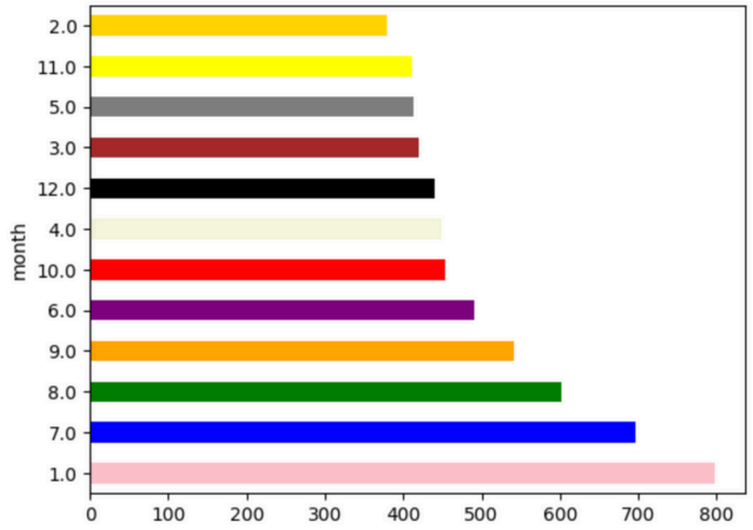
- **df["country"].value_counts([:5].plot.barh(color=colors)** to see which countries had the most attacks. We also did it for activity, gender, type...



Exploratory data analysis methods used & insights

```
colors = ['pink', 'blue', 'green', 'orange', 'purple',"red","beige","black","brown","grey","yellow","gold"]
df["month"].value_counts().plot.barh(color=colors)

#from December to February is summer in Australia. Maybe that is why january has a lot of attacks
#august, june and july and in the top 4
#in general, the best surfing conditions can be found during winter months
```

```
<Axes: ylabel='month'>
```



We discovered with df.info that date was an object, so we investigated the column with **print(df['date'])**:

```
print(df['date']) #different data type entries
```

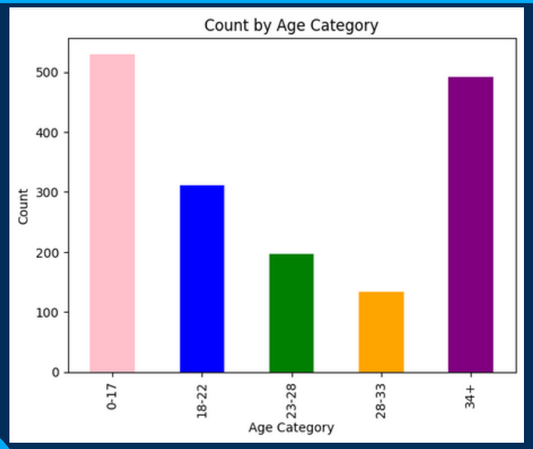
```
0       2024-09-16 00:00:00
1       2024-08-26 00:00:00
2       2024-08-06 00:00:00
3       2024-07-23 00:00:00
4       2024-07-18 00:00:00
                ...
6965            Before 1903
6966            Before 1903
6967              1900-1905
6968              1883-1889
6969              1845-1853
Name: date, Length: 6967, dtype: object
```

```
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df['month'] = df['date'].dt.month
df.head(5)
```

4. INSIGHTS & CONCLUSIONS

Validated hypotheses

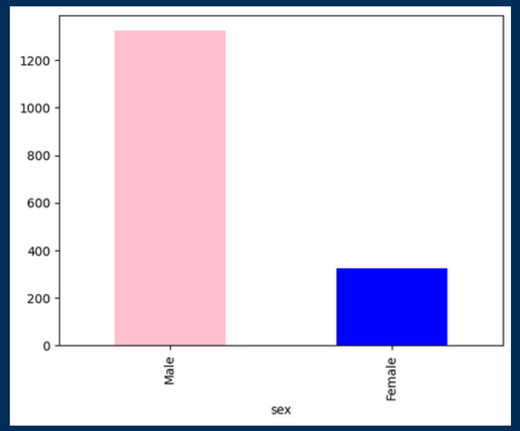
01 Young people suffer more shark attacks



Los más jóvenes (menores de edad) son los que más accidentes han declarado sufrir por tiburones, seguidos de los más mayores (34+)

La media de edad de atletas está en 27.

02 Men are more prone to shark attacks

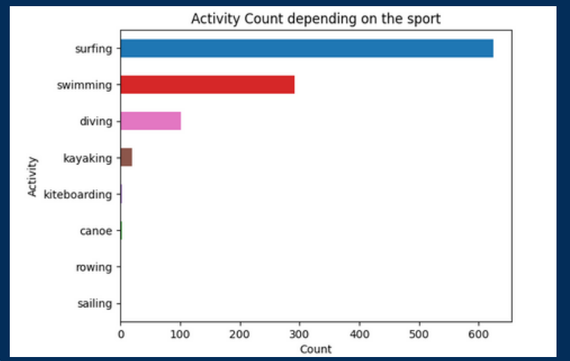


Los hombres han sufrido a lo largo de los datos registrados más accidentes por ataques de tiburones que las mujeres.

03 Coastal+warm states experience more shark attacks

En los estados de costa de Estados Unidos han sido las zonas donde más accidentes se han registrado.

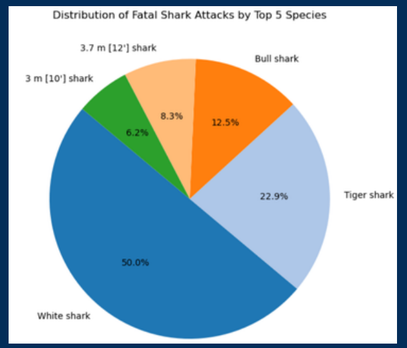
04 Athletes who practice surfing are at a higher risk



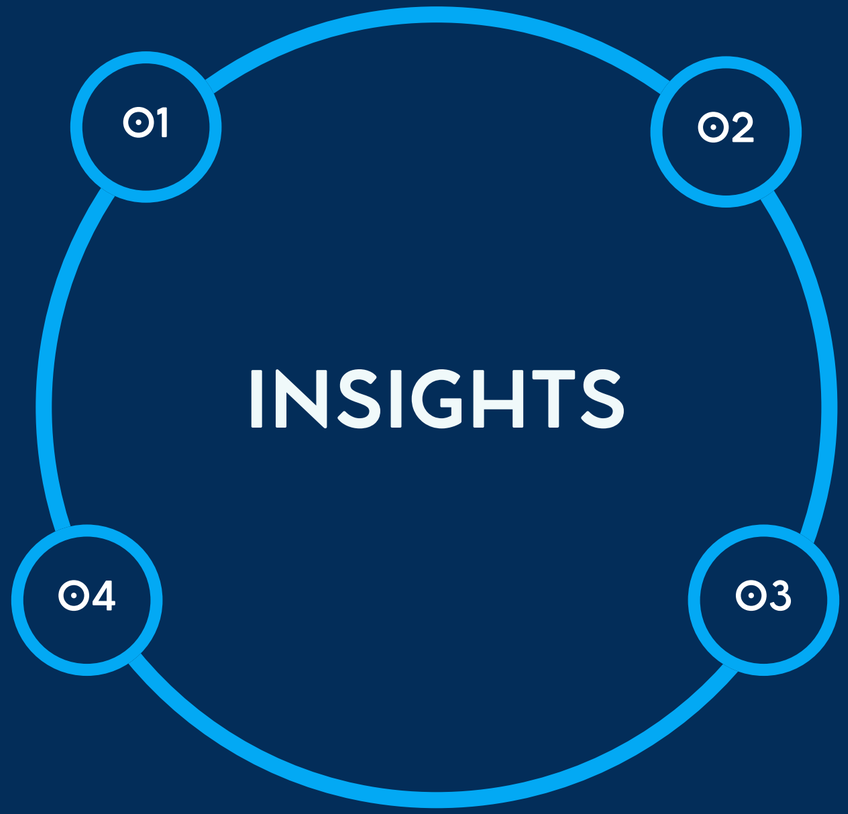
De las posibles actividades acuáticas, el surf es la más accidentes presenta

05 Bonus: White Shark

El tiburón blanco el que más ataques fatales provoca



PRICES
(according to our hypotheses)



Género

Los hombres tendrán un seguro de vida más caro que las mujeres.

79.65%
de los ataques registrados pertenecen a hombres

Edad

La media de edad de los deportistas está en 27 años. Esta franja de edad no es la que más accidentes ha sufrido.

El seguro de vida será más caro entre aquellos menores de edad y los mayores de 34 años.

Actividad

37.56%
de los incidentes registrados ocurrieron surfeando

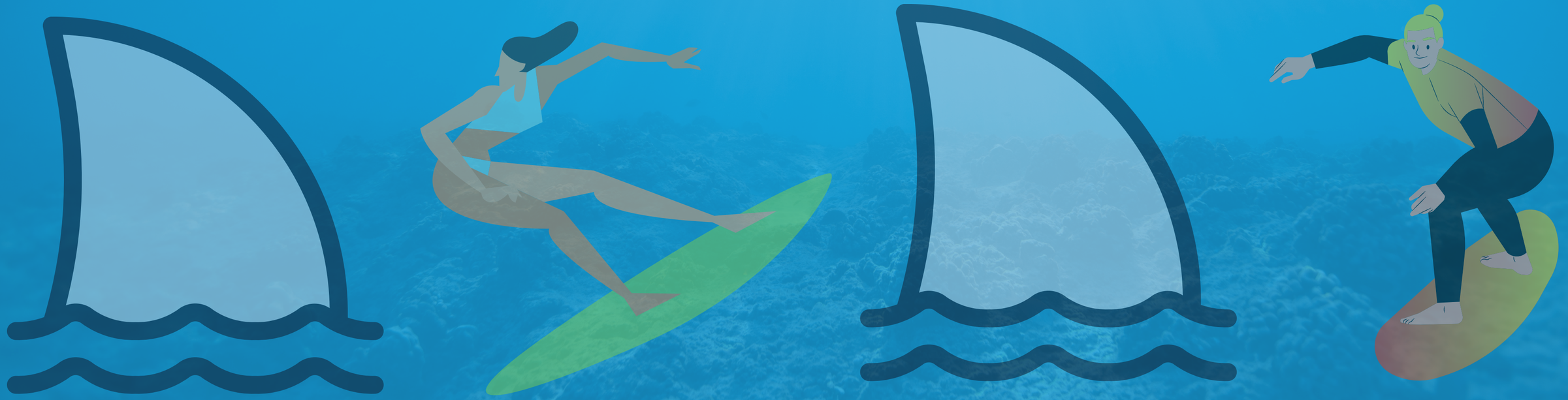
De los deportes celebrados en las Olimpiadas, aquellos que practiquen surf tendrán un precio más alto, seguido por natación y buceo.

Ubicación

97,5%
de los ataques registrados fueron en estados costeros.

Por ello, los deportistas que compitan en **Florida** (54%), **California** (12,7%) y **Hawaii** (10,2%) tendrán un seguro más caro

5. OBSTACLES AND LEARNINGS



5. MAJOR OBSTACLES AND LEARNINGS

Obstacle 1

A lot of time is spent in wrangling and cleaning, before you can start working with the data

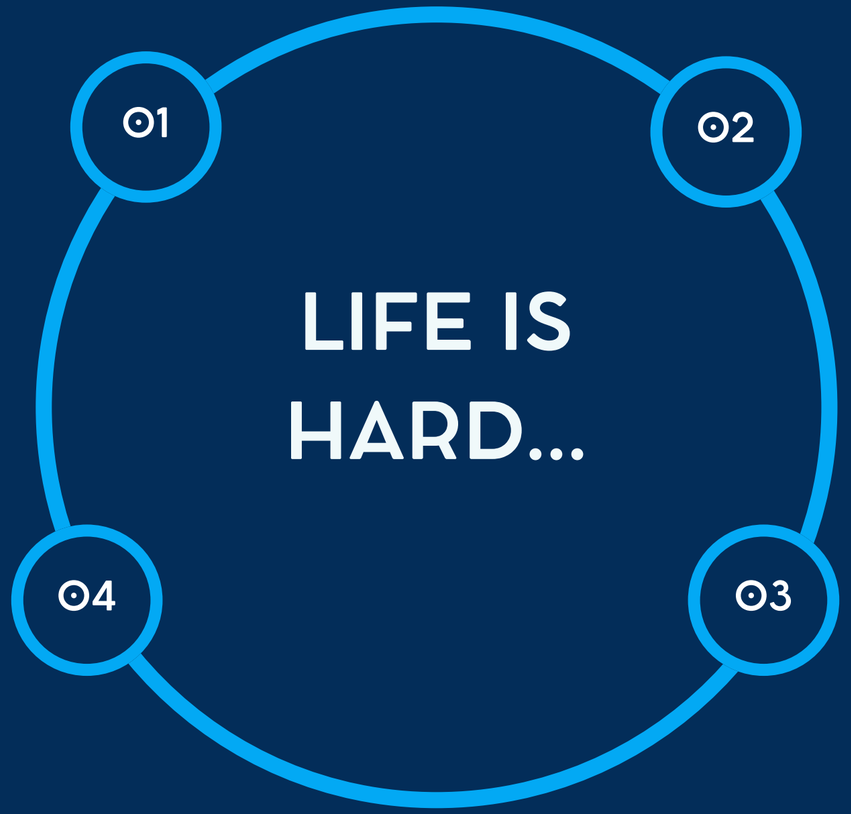
TIME

Obstacle 2

We started with a higher number of hypothesis and variables

COMPLEXITY

But we had to reduce them due to the lack of time



Learning 1

TASK DIVISION + TEAMWORK

Learning 2

SIMPLE IS BETTER

Our Team

Laura Sánchez
Lemon Shark

Benjamín Mancera
White Shark

Irene Gauna
Blue shark

Clara Gallego
Zambesi Shark

Miqueas Molina
Tiger Shark

THANK YOU!

Shark attacks insurance
Olympic games edition

Reggae Sharks| *Laura Sánchez, Benjamín Mancera, Irene Gauna, Clara Gallego, Miqueas Molina*