**Concordia Data Science Bootcamp Final Project**

**Here is the repository:** [**https://github.com/Anshuboom/Anshuboom\_SAFP\_repo**](https://github.com/Anshuboom/Anshuboom_SAFP_repo)

For convenience, the project proposal is left at the bottom of this doc as reference.

**Introduction**

The final project attempts to use ISAF compiled data (International Shark Attack Files) in order to do the following:

1. Predict the Fatality of a person and the probability of it, given that they were victims of a shark attack.
2. Perform a time series analysis of all attacks that have been recorded on file since the beginning.
3. Attempt to identify the culprit sharks based on their habitat, proximity to the user and historical behavior.
4. Chart and visualize the composition of the gathered data with respect to attacks by shark, by country and in the oceans, as well as the calculated risk they pose if attacked by them.
5. Allow the user to enter the relevant new (X record) data in a form, predict the fatality outcome, and plot the new user on a map along with the closed proximity attacks.

So it attempts to answer the following questions:

1. What is the global trend of shark attacks in the world, is it truly increasing as rumors and media sensationalization seem to be alluding to?
2. What countries have the most attacks and by what sharks?
3. Which sharks are the most ‘dangerous’ ones?
4. Given my location, my gender, timeslot, and activity I am performing in the water, if I do get attacked by a shark, will it be fatal and what is the probability of fatality?

DISCLAIMER:

The model suggested below is meant to be an academic exercise, not a prediction model to be taken literally. There are a few important assumptions that have been made to get there.

1. It does not take into account the shark population density in a given location
2. It does not take into account the migratory patterns of living sharks across the oceans. Lately Great White attacks and sightings in South Africa have dropped DRAMATICALLY, it is unclear why and historical evidence will be biased
3. Only 30% of all attacks have sharks as identified, out of those most witnesses exaggerate the sightings and seem to point at Great Whites. There are MANY other sharks in the sea and miss identifications are common.
4. Many Fatalities are assumed to be sharks in the dataset they could have been other factors involved.
5. The shark suggester is based on known sharks in the area and has been ordered (painstakenly) according to their occurrence in those waters example Greenland sharks will always appear last in the polar list but that is just historical. Water temperatures are changing, food sources are dwindling, sharks are adaptable and move, and as such we tend to see cooler sharks in warmer waters like the blue shark and warmer sharks in cooler waters like the bull shark. By and large, this data is still noisy and it will take many years to get better quality of records.
6. Activities that bait the sharks or provoke the sharks to appear like chumming used frequently in eco-tourism are not considered a fair depiction of their behavior.

All this said, to me, some sharks are indeed dangerous, they are carnivorous predators that will eat when hungry, and many are opportunistic feeders in an environment where food is getting scarcer. If there have been attacks around an area, to me, clearly, precautions need to be taken for it cannot be assumed that sharks are not curious enough to approach, get excited based on stimuli, bump, then attack if territorial instincts kick in and then even kill humans accidentally or intentionally, especially if we encroach on their territory more and more often. They are formidable creatures to say the least, an old, old design (much older than dinosaurs) evolved in many, many adaptations that are still classic. elegant and awe inspiring. Sharks are not dolphins 😊

**The Sharks picked in my ‘dangerous’ sharks set:**

https://en.wikipedia.org/wiki/Blacktip\_shark

https://en.wikipedia.org/wiki/Blue\_shark

https://en.wikipedia.org/wiki/Bronze\_whaler

https://en.wikipedia.org/wiki/Bull\_shark

https://en.wikipedia.org/wiki/Dusky\_shark

https://en.wikipedia.org/wiki/Gal%C3%A1pagos\_shark

https://en.wikipedia.org/wiki/Goblin\_shark

https://en.wikipedia.org/wiki/Grey\_nurse\_shark

https://en.wikipedia.org/wiki/Grey\_reef\_shark

https://en.wikipedia.org/wiki/Hammerhead\_shark

https://en.wikipedia.org/wiki/Lemon\_shark

https://en.wikipedia.org/wiki/Mako\_shark

https://en.wikipedia.org/wiki/Nurse\_shark

https://en.wikipedia.org/wiki/Oceanic\_whitetip\_shark

https://en.wikipedia.org/wiki/Porbeagle\_shark

https://en.wikipedia.org/wiki/Raggedtooth\_shark

https://en.wikipedia.org/wiki/Sand\_tiger\_shark

https://en.wikipedia.org/wiki/Sandbar\_shark

https://en.wikipedia.org/wiki/Sevengill\_shark

https://en.wikipedia.org/wiki/Silky\_shark

https://en.wikipedia.org/wiki/Spinner\_shark

https://en.wikipedia.org/wiki/Thresher\_shark

https://en.wikipedia.org/wiki/Tiger\_shark

https://en.wikipedia.org/wiki/Unknown\_shark

https://en.wikipedia.org/wiki/Great\_white\_shark

https://en.wikipedia.org/wiki/Whitetip\_reef\_shark

https://en.wikipedia.org/wiki/Wobbegong\_shark

<https://en.wikipedia.org/wiki/Zambezi_shark>

**My Data and wrangling it**

Two notebooks were used in general to clean, input and format the data wrangler.ipynb, wrangler2.ipynb. The data columns I was dealing with were

['Case Number', 'Date', 'Year', 'Type', 'Country', 'Area', 'Location',

'Activity', 'Name', 'Sex ', 'Age', 'Injury', 'Fatal (Y/N)', 'Time',

'Species ', 'Investigator or Source', 'pdf', 'href formula', 'href',

'Case Number.1', 'Case Number.2', 'original order'],

The challenges were:

1. Date column which was going to be the index of my time series was in string format and had multiple patterns – using regex
2. The year columns was also a string and was not always 4 digits
3. Location was in 3 parts which I needed to concatenate in order to have google geolocation be able to give me the longitudes and latitudes for each
4. The ‘Time’ column was a complete mess, it was user input and had no formatting so I could get values like 9h40, 9am, 9AM, 9A.M. 21:00, nine o’clock: and had up to 30 % missing values. It took me a couple of days just to deal with this, I finally decided to extract just the hour in each case and map it to and easily one hot encodable set [dawn, morning, midday, afternoon, dusk, evening]. For values I did not have, I did a random selection between dawn, afternoon, dusk, night because I read in various sources that attacks tended to occur in the mornings, afternoons and evenings where light was low since attacks are generally at the surface of the water. This provided me with a somewhat normal distribution of time slots which correlated with the data that was present in the first place.
5. My next challenge was extracting the known sharks from the ‘species’ to put the neatly into the SharkSpecies column. This column was once again user input and had all kinds of strings for the same species, 9ftGreatwhite, Great White, White, greatwhite, GW, gw. I used a variety of regexes to systematically extract the sharks.
6. The age column had to be eliminated because I only had 40% of the data. I tried to random select 30-60 but it still gave me spikes that strayed away from normality.
7. My most important column ended up being Activity and I nlp’d it because once again this was user input and sometimes had paragraphs, sometimes simple words like ‘Diving’ This ended up being a great decision because it captured entries like ‘surviving after the boat capsized’ or ‘wading in waist deep water”. I went through the usual word cleaning and token extraction routines to final TFidF vectorise the words.
8. in the end I was left with dfAttacksX.csv provided in the data folder

**Feature Engineering / new feature creation, ‘risk’:**

For each record in my dataset now consisting of about 6600 rows, I computed a risk column which was based on the stats I compiled using the sharks, the locations and the probabilities prior based on the population. Each shark had a score as did each country each major sea (ex Gulf of Mexico), this was achieved bu compiling various sorted dictionaries with prior probability of fatality based on the attacks calculated using the 6600 records. This risk column was a float value between 1 and 10. This was also used as an X column in order to improve the accuracy of the model.

**Column Selection for my X**

As initially planned, I intended to use a pipeline which would include the column transformers and classifiers. Here is how the column transformation was organized with respect to feature transformation and selection:

categorical\_features = ['Country', 'Gender','Timeslot','Zone']

numeric\_features = ['latitude','longitude','risk']

drop\_features = ['CaseNumber', 'Date', 'Year', 'Type', 'Area', 'Location','Name', 'Injury','Time','Species', 'SharkSpecies', 'full\_location', 'pos', 'latitude\_rad','longitude\_rad']

word\_features = ['Activity']

**Time Series Analysis and Modeling**

Since I had the dates on every record, I wanted to perform a time series analysis. The first thing I did was group the ATTACKS by year and analyse that as a time series. I wanted to predict only the next 2 years.

**Chart, histogram

Description automatically generated**

After performing some analysis on this to see what kind of stationarising it needed it was clear that there was no seasonal trend, just what looked liked an non linearly increasing trend.

from statsmodels.tsa.seasonal import seasonal\_decompose

**Graphical user interface, application

Description automatically generated with medium confidence**

Performing auto\_arima I ended up with p,d,q values of 1,1,2

ARIMA(1,1,2)(0,0,0)[0] as the best model

The fact that it was just 1st order differencing was a bit of a surprise because the increasing trend appeared to be increasing in a non linear way.

The first-order differencing suggests that the original series may have had some trend or seasonality that was removed to make the series stationary.

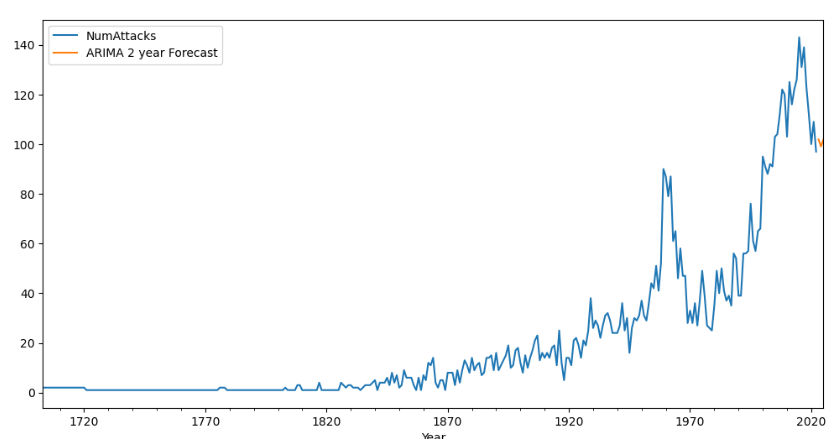
Based on this I was able to predict the Attack values for 2023, 2024 and 2025 as

2023-01-01 101.893624

2024-01-01 99.082357

2025-01-01 101.771209

**These values seemed to capture a downward trend suggesting that the attack numbers are not increasing compared to the previous few years.**

****

**Fatality (y/n) prediction, Model Choice**

The problem at hand was really a Binary Classification so 3 models were selected which were compared using accuracy scores and those were:

1. our trusty Logistical Regression and 2 more powerful tree based models
2. LGBM
3. XgBOOST

from sklearn.linear\_model import LogisticRegression

import lightgbm as lgb

import xgboost as xgb

**Model Selection and Hyper Parameter Tuning**

In each case I used GridSearchCV or RandomisedSearch CV on my pipeline with various parameters

Here is the pipeline definition:

preprocessor = ColumnTransformer(

transformers=[

('Categorical', OneHotEncoder(handle\_unknown='ignore'), categorical\_features),

# ('numerical', StandardScaler(), numeric\_features),

('numerical', PowerTransformer(), numeric\_features),

# ('text', CountVectorizer(tokenizer=lambda text: [tok.text for tok in nlp(text)]), 'Activity'),

('text', TfidfVectorizer(tokenizer=tokenize\_text), 'Activity'),

('droplist', 'drop', drop\_features)

],

remainder='passthrough'

)

# Define the pipeline

pipeline = Pipeline([

('preprocessor', preprocessor),

('classifier', LogisticRegression())

]

**For the preprocessor, I compared:**

**StandardScaler** and **PowerTransformer** – both of which scale features and normalize them

I also compared TFidf and CountVectorisers

**For the models I compared the accuracy scores of the following estimators, optimized**

**Logistic Regression**

**param\_grid = {}**

**param\_grid['penalty'] = ['l1', 'l2', 'none']**

**param\_grid['C'] = [0.005, 0.05, 0.5, 5, 50, 500]**

**param\_grid['solver'] = ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']**

{'classifier\_\_solver': 'liblinear', 'classifier\_\_penalty': 'l1', 'classifier\_\_C': 5, 'classifier': LogisticRegression(C=5, penalty='l1', random\_state=1, solver='liblinear')}

0.7894736842105263

**Which gave me the beset accuracy score of 0.79 once I allowed C to a range between 1 and 15**

**For LGBM I randomSearched the following hyperParams**

**param\_gridLoglgbm = {}**

**param\_gridLoglgbm['classifier\_\_max\_depth'] = [3,5,7]**

**param\_gridLoglgbm['classifier\_\_learning\_rate'] = [0.01,0.1,1]**

**param\_gridLoglgbm['classifier\_\_n\_estimators'] = [100, 500, 1000]**

**param\_gridLoglgbm['classifier\_\_num\_leaves'] = [10, 20, 30]**

**param\_gridLoglgbm['classifier\_\_min\_child\_samples'] = [10,20,30]**

**param\_gridLoglgbm['classifier']=[lgbm]**

**lgbmclassifierparams = [param\_gridLoglgbm]**

**I got the following results:**

Best parameters found: {'classifier\_\_num\_leaves': 20, 'classifier\_\_n\_estimators': 1000, 'classifier\_\_min\_child\_samples': 20, 'classifier\_\_max\_depth': 7, 'classifier\_\_learning\_rate': 0.01, 'classifier': LGBMClassifier(learning\_rate=0.01, max\_depth=7, n\_estimators=1000,

num\_leaves=20)}

Best accuracy score found: 0.7491916996668042

**For xgboost I randomSearched the following hyperParams**

param\_gridLogxgb = {}

param\_gridLogxgb['classifier\_\_Learning\_rate'] = [0.03,0.3]

param\_gridLogxgb['classifier\_\_max\_depth'] = [10, 15, 20]

param\_gridLogxgb['classifier\_\_gamma'] = [0.1, 0.2, 0.3, 0.4]

param\_gridLogxgb['classifier\_\_reg\_alpha'] = [0, 0.001, 0.01]

param\_gridLogxgb['classifier\_\_scale\_pos\_weight'] = [1, 2, 5, 10]

param\_gridLogxgb['classifier\_\_n\_estimators'] = [200, 400, 800]

param\_gridLogxgb['classifier']=[xgClassifier]

xgbclassifierparams = [param\_gridLogxgb]

Best parameters found: {'classifier\_\_scale\_pos\_weight': 1, 'classifier\_\_reg\_alpha': 0.01, 'classifier\_\_n\_estimators': 800, 'classifier\_\_max\_depth': 15, 'classifier\_\_gamma': 0.4, 'classifier\_\_Learning\_rate': 0.03, 'classifier': XGBClassifier(Learning\_rate=0.03, base\_score=None, booster=None

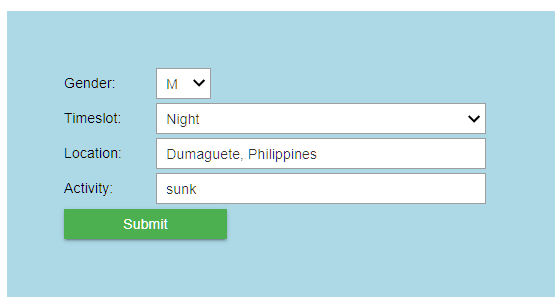
Best accuracy score found: 0.7387428278429546

**I picked LogisticRegression() as my winning estimator**

**I trained it again on my entire dataset to finalize it and saved it as trained\_pipeline\_LogisticRegression.pkl found in the Models folder.**

**The Final Product and how it is used:**

1. The user gets a form, to fill in Gender, TimeSlot, Location and Activity

****

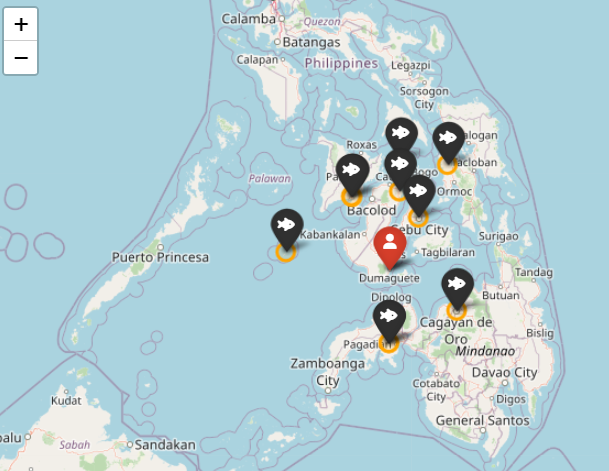
1. Location returns the longitude and latitude from which the country, sea, climatezone are determined.
2. Activity is nlp’d and can contain descriptions of the activity which give more specific probability predictions.
3. Once submitted, a full xrecord is created in the form of a dataframe to be fed into the pipeline
4. The prediction and prediction probability are returned:

Location country: Philippines

Risk associated to this Country: 1.8

NO, Attack MAY not be Fatal, with a predicted probability of: 0.3779994941786306

1. If probability exceeds 0.5, he prediction is yes otherwise it is no
2. Once Fatality has been predicted, the location of the new X can be mapped. This location is then used for spatial data analysis using ball tree nearest neighbor to determine the 10 closest attack locations to the location spot along with their distances from the spot and a 20 km proximity circles to determine most likely sharks. Red spot is the current spot.



1. Each spot can be clicked on to reveal its location and distance
2. Finally, the shark picker can be run
3. The algorithm tries to see if the closest neighbors had identified sharks. If the red spot is within one or more orange circles that have identified sharks, a list of sharks is compiled, otherwise, the country, the sea or the temperature zone determines the most common predatory sharks in the area.

PHILIPPINES

Could not find any IDENTIFIED sharks that have actually been IMPLICATED in attacks within 20 km of your location.

These are however risky sharks living in the waters of your location that could pose a threat if you were attacked.

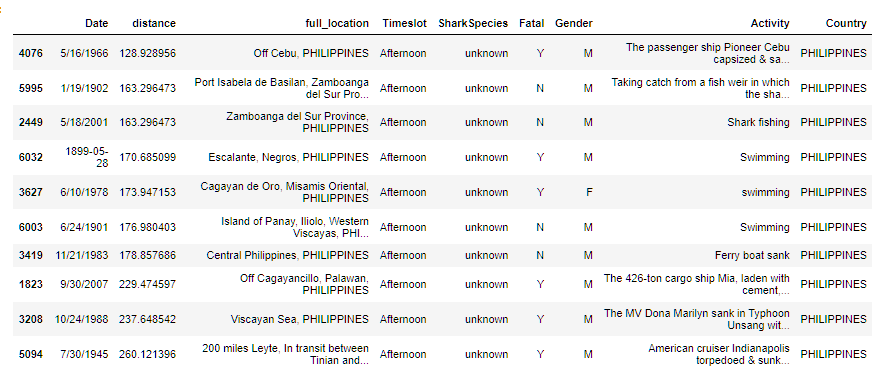
['oceanic whitetip', 'grey reef', 'hammerhead', 'bull', 'white shark', 'tiger shark', 'blacktip']

Out of these sharks it charts the probability of fatality (danger) each shark poses:

Chart, bar chart

Description automatically generated

1. Finally it also prints out the subset of the dataframe with the neighbours and the activity that occurred there:



**So what next?**

There are 2 things I want to do in my own time.

1. Use a NN or CNN model to dee if I can get an accuracy of better than 80%
2. Make this available via API that a website can consume.

=====================================================================================

**Concordia Data Science Bootcamp Final Project Proposal**

Shark attack Analysis and Recommendation

**Goal:**

This project will try to answer the following question:

If I am <activity: Swimming, Snorkeling, Surfing at <location: thinking, country with location name> at <time> with water conditions <temp, visibility>, if I am attacked, what kind of shark could it be and will it be fatal?

**Data**

I have already gathered some data from various sites: here is a snippet. I need to clean, enrich and impute a lot of information to make this usable. I will need to scrape some information from:

[Shark Attacks by Country (sharkattackdata.com)](http://www.sharkattackdata.com/place)

<https://www.seatemperature.org/>

[Global Shark Attack - World — Opendatasoft](https://public.opendatasoft.com/explore/dataset/global-shark-attack/api/?disjunctive.country&disjunctive.area&disjunctive.activity)

Graphical user interface, diagram, application

Description automatically generated with medium confidence

**Dimensions I want to consider in my X:**

1. Date: This data is a time series of actual recorded shark attacks across the world
2. Time of Day <dawn, mid-morning, afternoon, dusk, night> categorical
3. Location: Country, location which I will get the Long Lat for
4. Water Temperature
5. Activity
6. Injury – NLP

**What will the model(s) predict**

The Y will be the predicted fatality of the attack.

The Y could also be the classification of the shark species.

The Y could also be a prediction of whether this was an unprovoked – predatory attack or provoked based on activities like fishing chumming that are known to attract sharks.

**How I intend to do this:**

1. I want to see if there is a possibility that I could us ML techniques with Neural Nets to classify the sharks, but I don’t know if I have enough data you need to advise because this is what I really want to dive into.
2. Logistic regressions of various shapes seem obvious to me here.
3. I want to focus my attention on column transforming and pipeline building for my models to demonstrate that I have understood those concepts and can apply them
4. I don’t know how but in the end, there might be a form you fill with the help of dropdowns to get your answer with a blurb: saying:

In the case of an attack expect a Great White, injuries will be severe, and the chances of Fatality are high!