Prediction Tsunami

2. Data Understanding

- Collect Initial Data : เป็นขั้นตอนแรกในการดำเนินการเมื่อเริ่มต้นโปรเจคซึ่งการเก็บข้อมูลเข้าสู่โปรเจคของเรานั้น จะต้องดึงข้อมูลจากแหล่งที่เก็บข้อมูลไว้ เช่น Google Drive สามารถดำเนินการได้ดังนี้ :

1. Mount Google Drive: เชื่อมต่อ Google Drive เข้ากับ Google Colab เพื่อให้สามารถเข้าถึงไฟล์และโฟลเดอร์บน Google Drive ได้ โดยการใช้คำสั่ง drive.mount('/content/drive')

2. Import Libraries: ขั้นตอนนี้เรา import ไลบรารี pandas และ numpy เพื่อใช้ในการจัดการข้อมูลและการทำงานกับข้อมูลใน Python

3. Load Data: ด้วยการใช้คำสั่ง pd.read_csv() เพื่อโหลดข้อมูลจากไฟล์ CSV ที่อยู่ใน Google Drive โดยระบุที่อยู่ของไฟล์ CSV ที่เก็บไว้ในพารามิเตอร์ของ pd.read_csv() ซึ่งไฟล์ CSV นี้จะถูกโหลดเป็น DataFrame ด้วย pandas และเก็บไว้ในตัวแปร eq ที่เราสามารถใช้ในการทำงานกับข้อมูลได้ต่อไป โดยแสดงข้อมูล eq ดังรูปต่อไปนี้

	title	magnitude	date_time	cdi	mmi	alert	tsunami	sig	net	nst	dmin	gap	magType	depth	latitude	longitude	location	continent	country
0	M 7.0 - 18 km SW of Malango, Solomon Islands	7.0	22-11-2022 02:03	8	7	green	1	768	us	117	0.509	17.0	mwv	14.000	-9.7963	159.596	Malango, Solomon Islands	Oceania	Solomon Islands
1	M 6.9 - 204 km SW of Bengkulu, Indonesia	6.9	18-11-2022 13:37	4	4	green	0	735	us	99	2.229	34.0	mwv	25.000	-4.9559	100.738	Bengkulu, Indonesia	NaN	NaN
2	M 7.0 -	7.0	12-11-2022 07:09	3	3	green	1	755	us	147	3.125	18.0	mwv	579.000	-20.0508	-178.346	NaN	Oceania	Fiji
3	M 7.3 - 205 km ESE of Neiafu, Tonga	7.3	11-11-2022 10:48	5	5	green	1	833	us	149	1.865	21.0	mwv	37.000	-19.2918	-172.129	Neiafu, Tonga	NaN	NaN
4	M 6.6 -	6.6	09-11-2022 10:14	0	2	green	1	670	us	131	4.998	27.0	mwv	624.464	-25.5948	178.278	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
777	M 7.7 - 28 km SSW of Puerto El Triunfo, El Sal...	7.7	13-01-2001 17:33	0	8	NaN	0	912	us	427	0.000	0.0	mwc	60.000	13.0490	-88.660	Puerto El Triunfo, El Salvador	NaN	NaN
778	M 6.9 - 47 km S of Old Harbor, Alaska	6.9	10-01-2001 16:02	5	7	NaN	0	745	ak	0	0.000	0.0	mw	36.400	56.7744	-153.281	Old Harbor, Alaska	North America	NaN
779	M 7.1 - 16 km NE of Port-Olry, Vanuatu	7.1	09-01-2001 16:49	0	7	NaN	0	776	us	372	0.000	0.0	mwb	103.000	-14.9280	167.170	Port-Olry, Vanuatu	NaN	Vanuatu
780	M 6.8 - Mindanao, Philippines	6.8	01-01-2001 08:54	0	5	NaN	0	711	us	64	0.000	0.0	mwc	33.000	6.6310	126.899	Mindanao, Philippines	NaN	NaN
781	M 7.5 - 21 km SE of Lukatan, Philippines	7.5	01-01-2001 06:57	0	7	NaN	0	865	us	324	0.000	0.0	mwc	33.000	6.8980	126.579	Lukatan, Philippines	NaN	Philippines

782 rows × 19 columns

- Describe Data : ขั้นตอนที่ใช้สำหรับการอธิบายข้อมูลใน DataFrame เพื่อให้เรารู้จักข้อมูลมากขึ้นและเตรียมตัวสำหรับการทำงานกับข้อมูลในขั้นตอนถัดไป โดยดำเนินการดังนี้:

1. eq.info(): ขั้นตอนนี้ใช้เพื่อแสดงข้อมูลเกี่ยวกับ DataFrame ที่ช่วยในการตรวจสอบประเภทของข้อมูลในแต่ละคอลัมน์ และตรวจสอบว่ามีข้อมูลหายไป (missing data) หรือไม่

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 782 entries, 0 to 781
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   title      782 non-null    object 
 1   magnitude  782 non-null    float64
 2   date_time  782 non-null    object 
 3   cdi        782 non-null    int64  
 4   mmi        782 non-null    int64  
 5   alert      415 non-null    object 
 6   tsunami    782 non-null    int64  
 7   sig        782 non-null    int64  
 8   net        782 non-null    object 
 9   nst        782 non-null    int64  
 10  dmin       782 non-null    float64
 11  gap        782 non-null    float64
 12  magType    782 non-null    object 
 13  depth      782 non-null    float64
 14  latitude   782 non-null    float64
 15  longitude  782 non-null    float64
 16  location   777 non-null    object 
 17  continent  206 non-null    object 
 18  country    484 non-null    object 
dtypes: float64(6), int64(5), object(8)
memory usage: 116.2+ KB
```

สรุปจากภาพมีรายละเอียดดังนี้:

- จำนวนข้อมูล: มีทั้งหมด 782 แถว (entries) และ 19 คอลัมน์ (columns)

- รายชื่อคอลัมน์และจำนวนข้อมูลที่ไม่หายไป (Non-Null) ในแต่ละคอลัมน์

- ประเภทของข้อมูลใน DataFrame ประกอบไปด้วย float64 (6 columns), int64 (5 columns), และ object (8 columns)

2. eq.isnull().sum() / eq.shape[0] * 100: ขั้นตอนนี้คำนวณอัตราส่วนของข้อมูลที่หายไป (missing data) ในแต่ละคอลัมน์ ผลลัพธ์ที่ได้จะแสดงในรูปแบบของเปอร์เซ็นต์ของข้อมูลที่หายไปในแต่ละคอลัมน์

3. eq.columns: ขั้นตอนนี้ใช้เพื่อแสดงรายชื่อคอลัมน์ทั้งหมดใน DataFrame เพื่อทำความเข้าใจกับโครงสร้างของข้อมูลและชื่อคอลัมน์ที่ใช้ในการอ้างอิงข้อมูลในภายหลัง

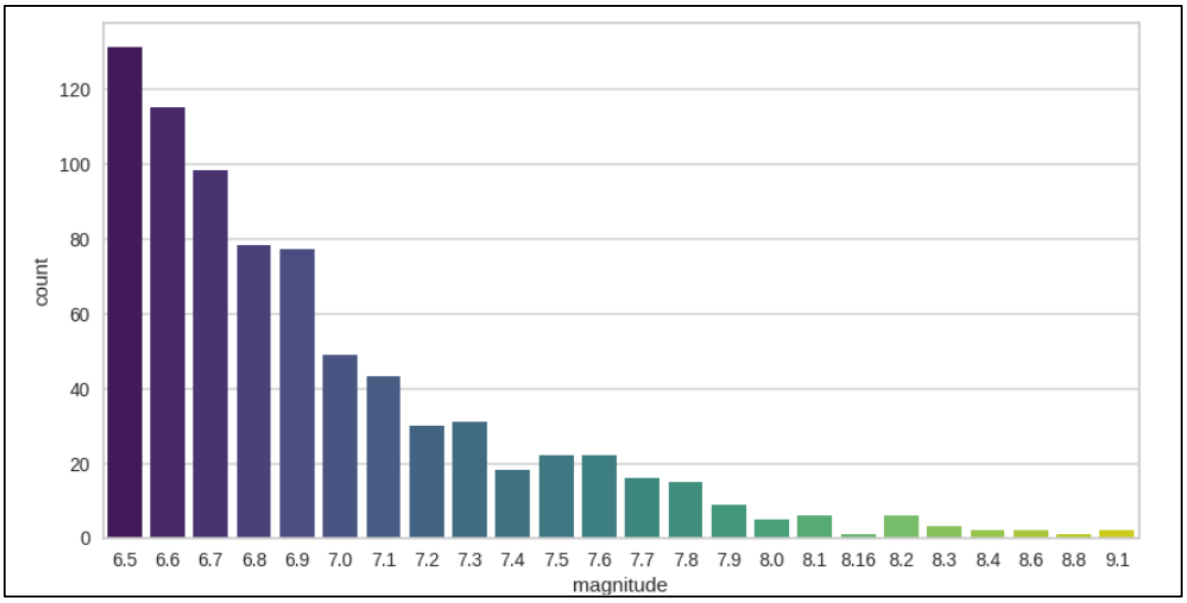
```
title          0.000000
magnitude      0.000000
date_time      0.000000
cdi            0.000000
mmi            0.000000
alert         46.930946
tsunami        0.000000
sig            0.000000
net            0.000000
nst            0.000000
dmin           0.000000
gap            0.000000
magType        0.000000
depth          0.000000
latitude       0.000000
longitude      0.000000
location       0.639386
continent     73.657289
country       38.107417
dtype: float64
```

อ้างอิงจากข้อ 2 แสดงเปอร์เซ็นต์ของข้อมูลที่หายไปในแต่ละคอลัมน์

```
Index(['title', 'magnitude', 'date_time', 'cdi', 'mmi', 'alert', 'tsunami',
       'sig', 'net', 'nst', 'dmin', 'gap', 'magType', 'depth', 'latitude',
       'longitude', 'location', 'continent', 'country'],
      dtype='object')
```

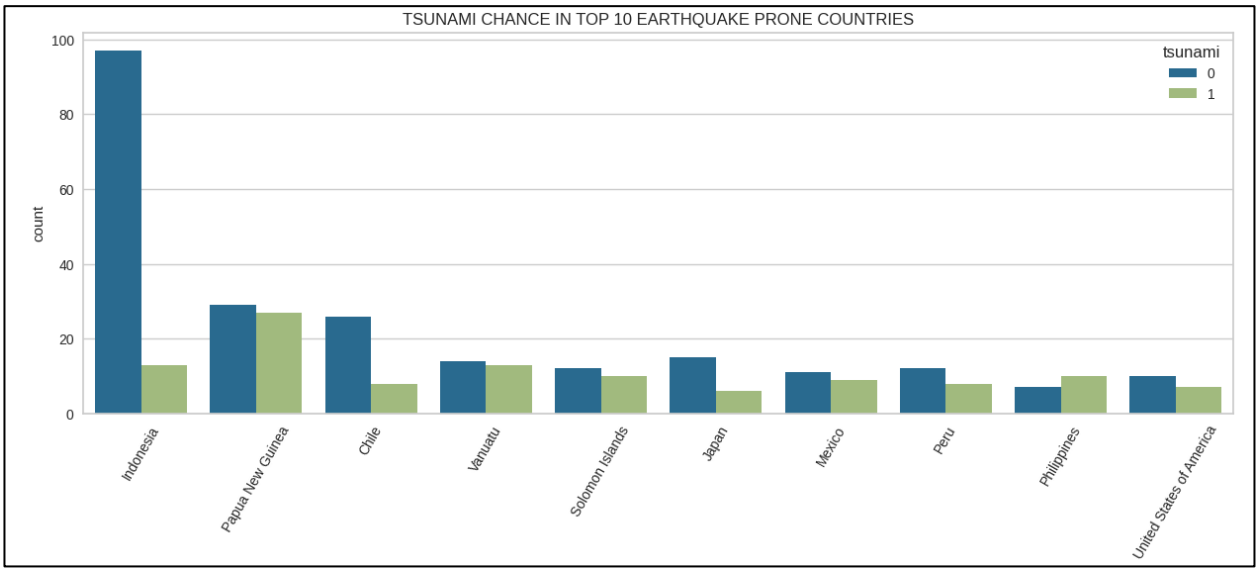
อ้างอิงจากข้อ 3 แสดงรายชื่อคอลัมน์ทั้งหมดใน DataFrame

- Explore Data : เป็นการสำรวจข้อมูลและสร้าง visualization เพื่อให้เราเข้าใจแนวโน้มของข้อมูลยิ่งขึ้น โดยใช้ ไลบรารี seaborn (sns) และ matplotlib.pyplot (plt) เพื่อใช้ในการสร้าง visualization ของข้อมูล รวมถึง %matplotlib inline เพื่อแสดงกราฟใน Jupyter Notebook เช่น
  - กราฟแท่งที่แสดงจำนวนของแต่ละระดับความรุนแรงของแผ่นดินไหว ('magnitude')



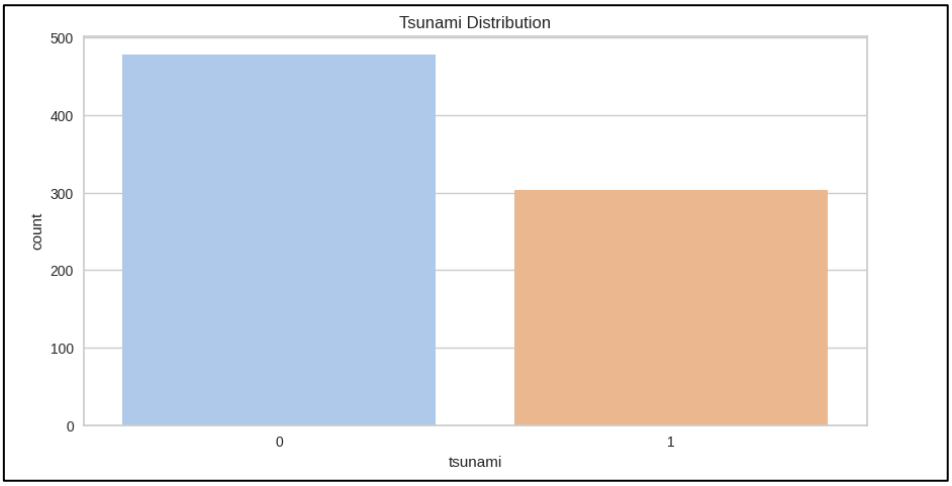
จากรูปภาพ ระดับความรุนแรงของแผ่นดินไหวที่ 6.5 magnitude มีจำนวนเยอะมากที่สุด และระดับความรุนแรงของแผ่นดินไหวที่ 9.1 magnitude มีจำนวนการเกิดขึ้นน้อยที่สุด

- กราฟแสดงการวิเคราะห์ว่าประเทศใดมีความเสี่ยงต่อการเกิดสึนามีเมื่อเกิดแผ่นดินไหว



จากรูปภาพ ประเทศอินโดนีเซียมีจำนวนการเกิดแผ่นดินไหวสูงที่สุดในโลก และประเทศปาปัวนิวกินีและฟิลิปปินส์นั้น มีความเสี่ยงที่จะเกิดสึนามิหลังจากที่แผ่นดินไหวเกิดขึ้นแล้ว

- กราฟแสดงการกระจายของข้อมูลการเกิดสึนามิ (tsunami) โดยใช้ Seaborn เพื่อสร้างกราฟ Countplot แสดงความถี่ของข้อมูลในคอลัมน์ 'tsunami' โดย ค่า 0 แทนไม่มีการเกิดสึนามีและ ค่า 1 แทนการเกิดสึนามิ



3. Data preparation

- Clean Data : ในขั้นตอน Load Data พบว่า ใน Dataframe มีค่า null ใน Columns ของ alert, continent, country เกิดขึ้น จึงทำการแทนที่ค่าว่างด้วยการใช้ eq.fillna และกำหนดค่าว่างของ Columns ดังนี้ 'alert': 'No Alert', 'continent': 'Unknown', 'country': 'Unknown', 'location': 'Unknown'

	title	magnitude	date_time	cdi	mmi	alert	tsunami	sig	net	nst	dmin	gap	magType	depth	latitude	longitude	location	continent	country
0	M 7.0 - 18 km SW of Malango, Solomon Islands	7.0	22-11-2022 02:03	8	7	green	1	768	us	117	0.509	17.0	mww	14.000	-9.7963	159.596	Malango, Solomon Islands	Oceania	Solomon Islands
1	M 6.9 - 204 km SW of Bengkulu, Indonesia	6.9	18-11-2022 13:37	4	4	green	0	735	us	99	2.229	34.0	mww	25.000	-4.9559	100.738	Bengkulu, Indonesia	NaN	NaN
2	M 7.0 -	7.0	12-11-2022 07:09	3	3	green	1	755	us	147	3.125	18.0	mww	579.000	-20.0508	-178.346	NaN	Oceania	Fiji
3	M 7.3 - 205 km ESE of Neiafu, Tonga	7.3	11-11-2022 10:48	5	5	green	1	833	us	149	1.865	21.0	mww	37.000	-19.2918	-172.129	Neiafu, Tonga	NaN	NaN
4	M 6.6 -	6.6	09-11-2022 10:14	0	2	green	1	670	us	131	4.998	27.0	mww	624.464	-25.5948	178.278	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
777	M 7.7 - 28 km SSW of Puerto El Triunfo, El Sal...	7.7	13-01-2001 17:33	0	8	NaN	0	912	us	427	0.000	0.0	mwc	60.000	13.0490	-88.660	Puerto El Triunfo, El Salvador	NaN	NaN
778	M 6.9 - 47 km S of Old Harbor, Alaska	6.9	10-01-2001 16:02	5	7	NaN	0	745	ak	0	0.000	0.0	mw	36.400	56.7744	-153.281	Old Harbor, Alaska	North America	NaN
779	M 7.1 - 16 km NE of Port-Olry, Vanuatu	7.1	09-01-2001 16:49	0	7	NaN	0	776	us	372	0.000	0.0	mwb	103.000	-14.9280	167.170	Port-Olry, Vanuatu	NaN	Vanuatu
780	M 6.8 - Mindanao, Philippines	6.8	01-01-2001 08:54	0	5	NaN	0	711	us	64	0.000	0.0	mwc	33.000	6.6310	126.899	Mindanao, Philippines	NaN	NaN
781	M 7.5 - 21 km SE of Lukatan, Philippines	7.5	01-01-2001 06:57	0	7	NaN	0	865	us	324	0.000	0.0	mwc	33.000	6.8980	126.579	Lukatan, Philippines	NaN	Philippines

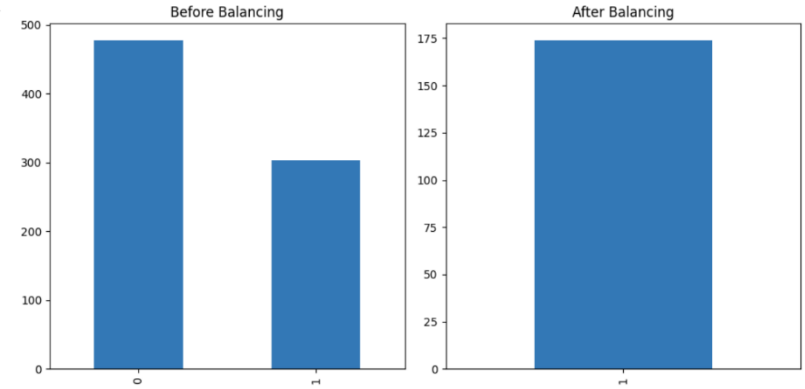
782 rows × 19 columns

หลังจากกำหนดค่าว่างเรียบร้อยแล้ว ตรวจสอบจำนวนข้อมูลที่มีค่าเป็น null ในแต่ละคอลัมน์ของ DataFrame 'eq' พบว่าไม่มีค่า null ในแต่ละคอลัมน์เลย ทุกคอลัมน์มีค่าข้อมูลที่ครบถ้วน ตามรูปภาพดังนี้

	magnitude	cdi	mmi	alert	tsunami	sig	net	nst	dmin	gap	magType	depth	latitude	longitude	location	continent	country
0	7.0	8	7	green	1	768	us	117	0.509	17.0	mww	14.000	-9.7963	159.596	Malango, Solomon Islands	Oceania	Solomon Islands
1	6.9	4	4	green	0	735	us	99	2.229	34.0	mww	25.000	-4.9559	100.738	Bengkulu, Indonesia	Unknown	Unknown
2	7.0	3	3	green	1	755	us	147	3.125	18.0	mww	579.000	-20.0508	-178.346	Unknown	Oceania	Fiji
3	7.3	5	5	green	1	833	us	149	1.865	21.0	mww	37.000	-19.2918	-172.129	Neiafu, Tonga	Unknown	Unknown
4	6.6	0	2	green	1	670	us	131	4.998	27.0	mww	624.464	-25.5948	178.278	Unknown	Unknown	Unknown
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
777	7.7	0	8	No Alert	0	912	us	427	0.000	0.0	mwc	60.000	13.0490	-88.660	Puerto El Triunfo, El Salvador	Unknown	Unknown
778	6.9	5	7	No Alert	0	745	ak	0	0.000	0.0	mw	36.400	56.7744	-153.281	Old Harbor, Alaska	North America	Unknown
779	7.1	0	7	No Alert	0	776	us	372	0.000	0.0	mwb	103.000	-14.9280	167.170	Port-Olry, Vanuatu	Unknown	Vanuatu
780	6.8	0	5	No Alert	0	711	us	64	0.000	0.0	mwc	33.000	6.6310	126.899	Mindanao, Philippines	Unknown	Unknown
781	7.5	0	7	No Alert	0	865	us	324	0.000	0.0	mwc	33.000	6.8980	126.579	Lukatan, Philippines	Unknown	Philippines

782 rows × 17 columns

- Construct Data : ปรับความสมดุลของคลาสในการสร้างโมเดล โดยการทำ balancing ข้อมูลในคอลัมน์ 'tsunami' เพื่อให้ข้อมูลสมดุลกัน และสร้างกราฟเพื่อเปรียบเทียบก่อนและหลังทำ balancing ได้ดังนี้



จากรูปภาพจำนวนข้อมูลก่อนการทำ balancing ของ 0 มีอยู่จำนวน 478 ตัว และข้อมูลของ 1 อยู่จำนวน 304 ตัว จึงทำการ balancing โดยจำนวนข้อมูลหลังการทำ balancing ของ 1 เพิ่มขึ้นมาเป็น 174 ตัว ทำให้ข้อมูลทั้ง 0 และ 1 มีจำนวนเท่ากัน และนำไปใช้ในขั้นตอนต่อไป

- Feature Engineering (format data) : ขั้นตอนต่อไปเป็นการแปลงข้อมูลที่มีรูปแบบเป็นข้อความให้กลายเป็นตัวเลข ซึ่งได้ทำการแปลงข้อมูลใน Columns ของ alert , net , magType , Location , continent และ country โดยใช้เทคนิค Label Encoding เพื่อให้สามารถนำไปประมวลผลในแบบโมเดลทางสถิติหรือแบบประมาณค่าได้โดยง่าย และนำค่าที่ได้มาใส่ในตารางข้อมูล ดังนี้

- ก่อนการแปลงข้อมูลใน Columns ของ alert , net , magType , Location , continent และ country

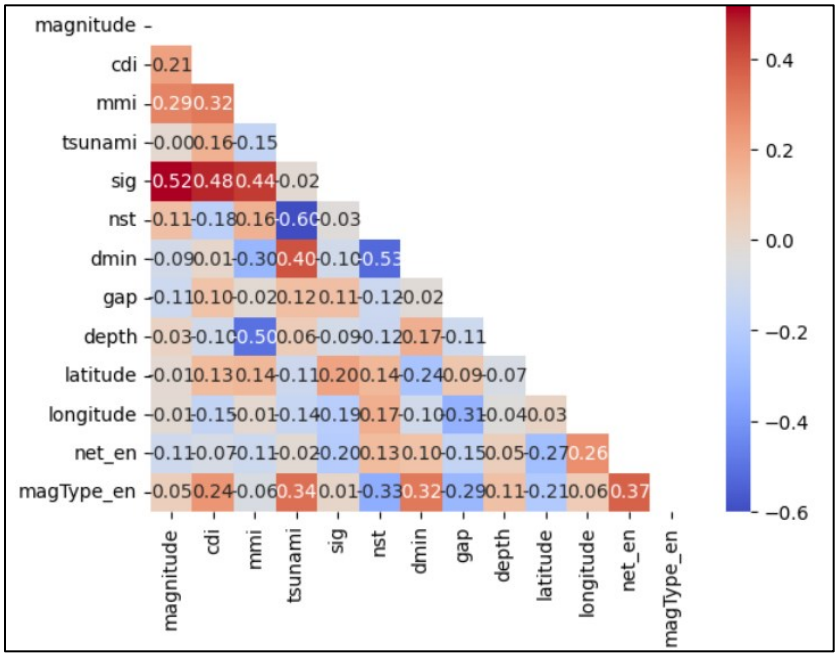
	magnitude	cdi	mmi	alert	tsunami	sig	net	nst	dmin	gap	magType	depth	latitude	longitude	location	continent	country
0	7.0	8	7	green	1	768	us	117	0.509	17.0	mww	14.000	-9.7963	159.596	Malango, Solomon Islands	Oceania	Solomon Islands
1	6.9	4	4	green	0	735	us	99	2.229	34.0	mww	25.000	-4.9559	100.738	Bengkulu, Indonesia	Unknown	Unknown
2	7.0	3	3	green	1	755	us	147	3.125	18.0	mww	579.000	-20.0508	-178.346	Unknown	Oceania	Fiji
3	7.3	5	5	green	1	833	us	149	1.865	21.0	mww	37.000	-19.2918	-172.129	Neiafu, Tonga	Unknown	Unknown
4	6.6	0	2	green	1	670	us	131	4.998	27.0	mww	624.464	-25.5948	178.278	Unknown	Unknown	Unknown
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
777	7.7	0	8	No Alert	0	912	us	427	0.000	0.0	mwc	60.000	13.0490	-88.660	Puerto El Triunfo, El Salvador	Unknown	Unknown
778	6.9	5	7	No Alert	0	745	ak	0	0.000	0.0	mw	36.400	56.7744	-153.281	Old Harbor, Alaska	North America	Unknown
779	7.1	0	7	No Alert	0	776	us	372	0.000	0.0	mwb	103.000	-14.9280	167.170	Port-Olry, Vanuatu	Unknown	Vanuatu
780	6.8	0	5	No Alert	0	711	us	64	0.000	0.0	mwc	33.000	6.6310	126.899	Mindanao, Philippines	Unknown	Unknown
781	7.5	0	7	No Alert	0	865	us	324	0.000	0.0	mwc	33.000	6.8980	126.579	Lukatan, Philippines	Unknown	Philippines

782 rows × 17 columns

- หลังทำการแปลงข้อมูลใน Columns ให้เป็น alert_en , net_en , magType_en , Location_en , continent_en และ country_em

alert_en	net_en	magType_en	location_en	continent_en	country_en
1	9	8	212	4	38
1	9	8	48	6	47
1	9	8	390	4	13
1	9	8	253	6	47
1	9	8	390	6	47

- Select data (ตาราง Correlation matrix) : การดึงความสัมพันธ์ระหว่างตัวแปรที่เกี่ยวข้องกับ 'tsunami' ออกมา โดยคำสั่งที่ใช้คือ correlation = eq.corr(): เพื่อสร้าง correlation matrix ของชุดข้อมูล eq ซึ่งจะเป็นตารางที่แสดงค่าสหสัมพันธ์ระหว่างคู่ของตัวแปรทุกคู่ในชุดข้อมูล ผลลัพธ์ที่ได้จากตารางนี้ คือ ซึ่งเป็นค่าสหสัมพันธ์ที่มีความสัมพันธ์กับ 'tsunami' มากที่สุด ดังรูปภาพ



และจากรูปภาพดังกล่าว ค่าสหสัมพันธ์ที่สูงสุด 5 ค่าที่เกี่ยวข้องกับตัวแปร 'tsunami' สรุปได้ดังนี้

Top 5 highest correlations with 'tsunami':

Column	**Correlations**
nst	0.600231
dmin	0.400752
magType_en	0.340445
cdi	0.160266
mmi	0.147363

4. Modeling

- Select modeling technique : เป็นขั้นตอนที่ใช้ Pycaret เพื่อเลือกเทคนิคหรือโมเดลที่เหมาะสมสำหรับปัญหาการจำแนกประเภท โดยสร้างและเปรียบเทียบโมเดลหลายๆ รูปแบบ เพื่อเลือกโมเดลที่มีประสิทธิภาพสูงสุดสำหรับชุดข้อมูล มีขั้นตอนดังนี้

  1. ติดตั้ง Pycaret: โดยใช้คำสั่ง !pip install pycaret เพื่อติดตั้ง Pycaret ในเซลล์โค้ดของ Google Colab.

  2. Import Pycaret Classification: ใช้คำสั่ง from pycaret.classification import * เพื่อโหลดฟังก์ชันและอ็อบเจกต์ที่จำเป็นสำหรับการสร้างและประเมินโมเดลทางการจำแนกประเภท (classification) โดย Pycaret. ได้ผลลัพธ์ดังนี้

	Description	Value
0	Session id	123
1	Target	tsunami
2	Target type	Binary
3	Original data shape	(782, 13)
4	Transformed data shape	(782, 13)
5	Transformed train set shape	(547, 13)
6	Transformed test set shape	(235, 13)
7	Numeric features	12
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fold Generator	StratifiedKFold
13	Fold Number	5
14	CPU Jobs	-1
15	Use GPU	False
16	Log Experiment	False
17	Experiment Name	clf-default-name
18	USI	a2fd

3. การเตรียมข้อมูล (Setup): ใช้ฟังก์ชัน setup() เพื่อเตรียมข้อมูลสำหรับการสร้างโมเดล โดยระบุข้อมูล (data) และตัวแปรเป้าหมาย (target) ที่ต้องการจำแนก และระบุ session_id เพื่อให้การเรียกใช้ฟังก์ชันทุกครั้งเริ่มต้นจากจุดเดียวกัน และ fold เพื่อกำหนดจำนวนของ fold ในการ cross-validation. ได้ผลลัพธ์ดังนี้

	Description	Value
0	Session id	123
1	Target	tsunami
2	Target type	Binary
3	Original data shape	(782, 13)
4	Transformed data shape	(782, 13)
5	Transformed train set shape	(547, 13)
6	Transformed test set shape	(235, 13)
7	Numeric features	12
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fold Generator	StratifiedKFold
13	Fold Number	5
14	CPU Jobs	-1
15	Use GPU	False
16	Log Experiment	False
17	Experiment Name	clf-default-name
18	USI	a84b

4. สร้างโมเดลที่เปรียบเทียบได้ (Compare Models): ใช้ฟังก์ชัน compare_models() เพื่อสร้างและเปรียบเทียบโมเดลจำนวนหลายๆ รูปแบบ เพื่อให้สามารถเลือกโมเดลที่เหมาะสมสำหรับข้อมูล

จากการทำ Compare Models ทำให้ได้ผลลัพธ์ 2 โมเดลที่จะนำมาใช้ทำนายการเกิดสึนามิ คือ Extreme Gradient Boosting (xgboost) และ Random Forest Classifier (rf) ที่มีประสิทธิภาพมากที่สุดในการทำนายข้อมูล

XGBClassifier

```
XGBClassifier(base_score=None, booster='gbtree', callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device='cpu', early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=-1,
              num_parallel_tree=None, objective='binary:logistic', ...)
```

- ภาพประกอบผลลัพธ์ที่ได้จากการ Compare Models โดย Extreme Gradient Boosting มีค่า Accuracy เท่ากับ 0.8903 และ Random Forest Classifier (rf) มีค่า Accuracy เท่ากับ 0.8885

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
xgboost	Extreme Gradient Boosting	0.8903	0.9456	0.8681	0.8537	0.8599	0.7698	0.7711	0.2820
rf	Random Forest Classifier	0.8885	0.9474	0.8588	0.8575	0.8570	0.7657	0.7671	0.5300
et	Extra Trees Classifier	0.8885	0.9406	0.8685	0.8491	0.8585	0.7665	0.7670	0.4340
gbc	Gradient Boosting Classifier	0.8867	0.9416	0.8590	0.8534	0.8548	0.7620	0.7637	0.4700
lightgbm	Light Gradient Boosting Machine	0.8848	0.9466	0.8493	0.8532	0.8500	0.7566	0.7580	1.1920
ada	Ada Boost Classifier	0.8739	0.9478	0.8216	0.8516	0.8355	0.7334	0.7346	0.5800
dt	Decision Tree Classifier	0.8556	0.8460	0.8027	0.8228	0.8116	0.6947	0.6959	0.0840
lda	Linear Discriminant Analysis	0.8245	0.8653	0.8734	0.7291	0.7944	0.6434	0.6521	0.0700
knn	K Neighbors Classifier	0.8227	0.8798	0.7934	0.7625	0.7760	0.6294	0.6317	0.1180
ridge	Ridge Classifier	0.8227	0.0000	0.8688	0.7280	0.7918	0.6393	0.6477	0.0820
lr	Logistic Regression	0.8135	0.8532	0.8501	0.7207	0.7797	0.6199	0.6269	2.2540
nb	Naive Bayes	0.8117	0.8272	0.8544	0.7170	0.7789	0.6170	0.6254	0.1060
qda	Quadratic Discriminant Analysis	0.7952	0.8411	0.8071	0.7075	0.7532	0.5793	0.5844	0.0860
svm	SVM - Linear Kernel	0.7659	0.0000	0.7346	0.5492	0.6280	0.4891	0.5075	0.0720
dummy	Dummy Classifier	0.6106	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0620

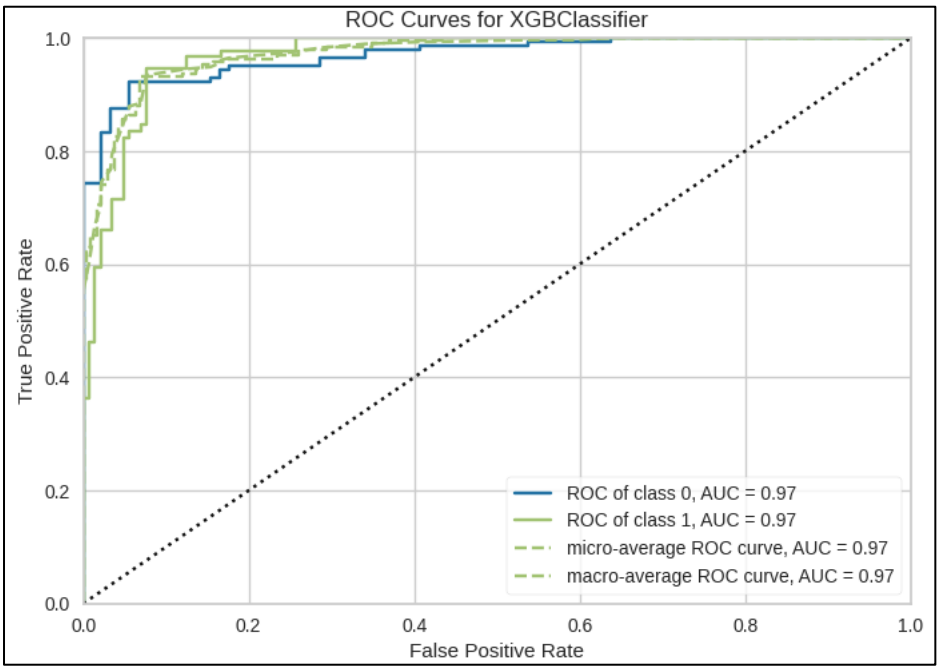
หลังจากนั้นจะทำการปรับค่าพารามิเตอร์ให้ดีที่สุด เริ่มจากโมเดลของ Extreme Gradient Boosting โดย

- ใช้คำสั่ง create_model('xgboost') เพื่อสร้างโมเดล XGBoost โดยใช้ค่าพารามิเตอร์เริ่มต้นที่ถูกกำหนดไว้ใน Pycaret โมดูล classification ด้วยการใช้ค่าพารามิเตอร์ที่เหมาะสมตามค่าเริ่มต้นของ Pycaret และข้อมูลที่กำหนด

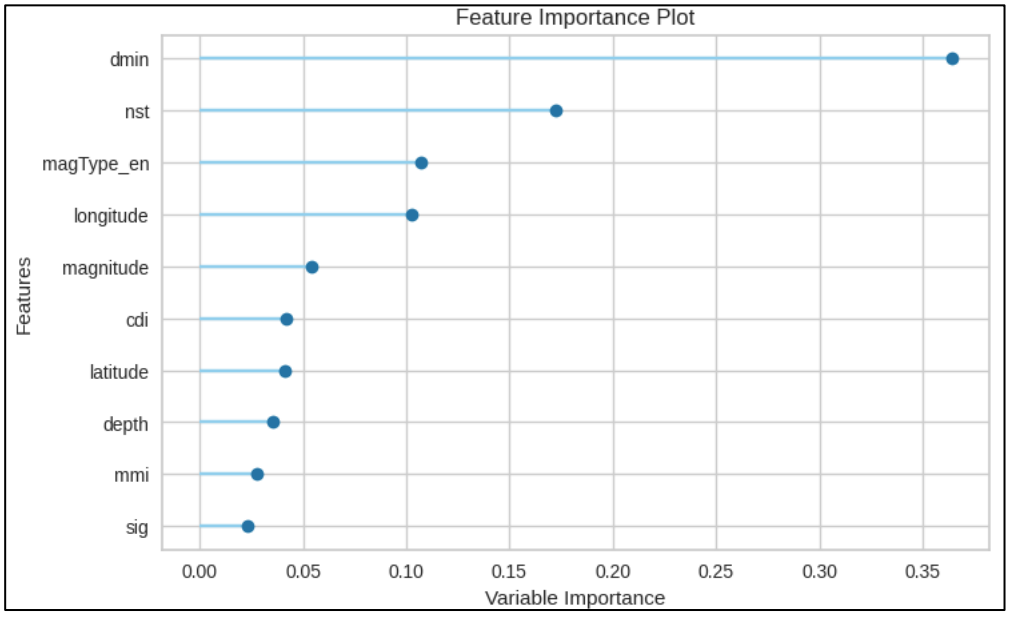
- คำสั่ง tune_model(xgboost_model) สำหรับปรับค่าพารามิเตอร์ของโมเดลเพื่อให้มีประสิทธิภาพมากที่สุด โดยใช้การค้นหาพารามิเตอร์ที่เหมาะสมโดยอัตโนมัติ ซึ่งจะทำการทดลองค่าพารามิเตอร์ต่างๆ และเลือกค่าที่ทำให้โมเดลมีประสิทธิภาพสูงสุด โดยใช้ cross-validation เพื่อประเมินประสิทธิภาพของโมเดลก่อนและหลังการปรับค่าพารามิเตอร์ เพื่อให้แน่ใจว่าการปรับค่าพารามิเตอร์นั้นมีประสิทธิภาพจริงๆ และไม่เกิดการ overfitting กับข้อมูลการทดสอบ ซึ่งจะได้ค่าพารามิเตอร์ที่เหมาะสม ดังนี้

```
XGBClassifier(base_score=None, booster='gbtree', callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device='cpu', early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=-1,
              num_parallel_tree=None, objective='binary:logistic', ...)
```

- เมื่อได้ค่าพารามิเตอร์แล้ว ในขั้นตอนต่อไป จะพล็อตกราฟที่เกี่ยวข้องกับประสิทธิภาพของโมเดล XGBoost ที่ถูกปรับค่าพารามิเตอร์แล้ว โดยใช้ plot_model(tuned_xgboost_model) แล้วจะแสดงกราฟ ROC Curve (Receiver Operating Characteristic Curve) ซึ่งเป็นกราฟที่แสดงความสัมพันธ์ระหว่าง True Positive Rate กับ False Positive Rate ซึ่งช่วยในการประเมินประสิทธิภาพของโมเดลในการแยกแยะคลาส



- ขั้นตอน plot_model ใช้คำสั่ง plot_model(tuned_xgboost_model, plot='feature') สำหรับการพล็อตกราฟที่เกี่ยวข้องกับคุณลักษณะ (features) หรือตัวแปรที่มีผลต่อการตัดสินใจของโมเดล XGBoost



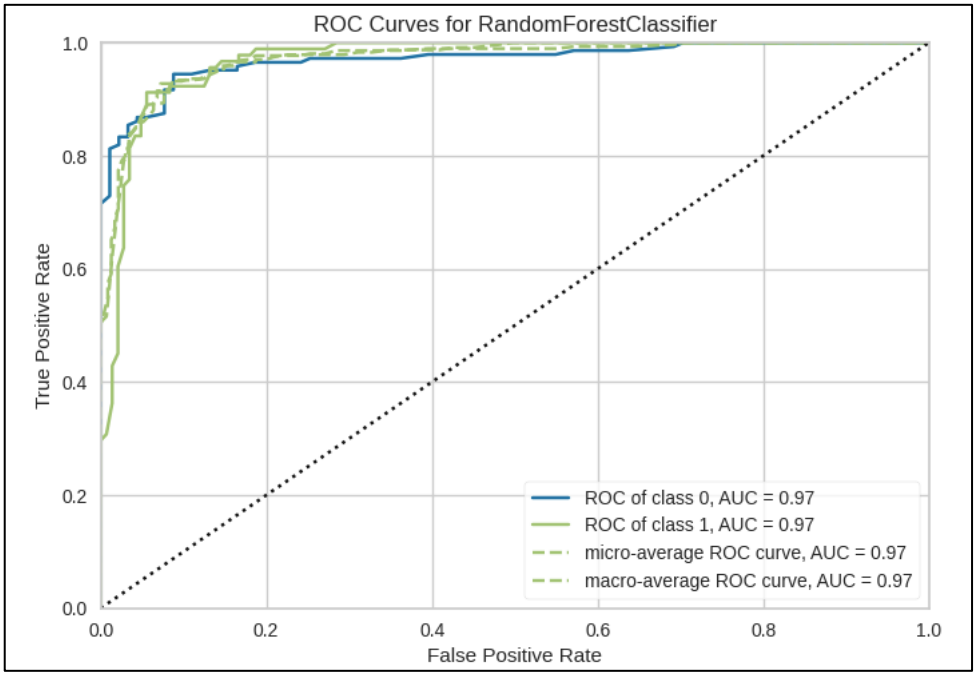
จากภาพ แสดง features ที่มีผลต่อการตัดสินใจของโมเดล XGBoost ใน 5 ลำดับแรก ดังนี้ dmin , nst, magType_en , longitude และ magnitude

ในส่วนของโมเดล Random Forest Classifier (rf) จะใช้วิธีเดียวกันเพื่อทำให้ได้ค่าพารามิเตอร์และ feature ที่เหมาะสม ดังนี้

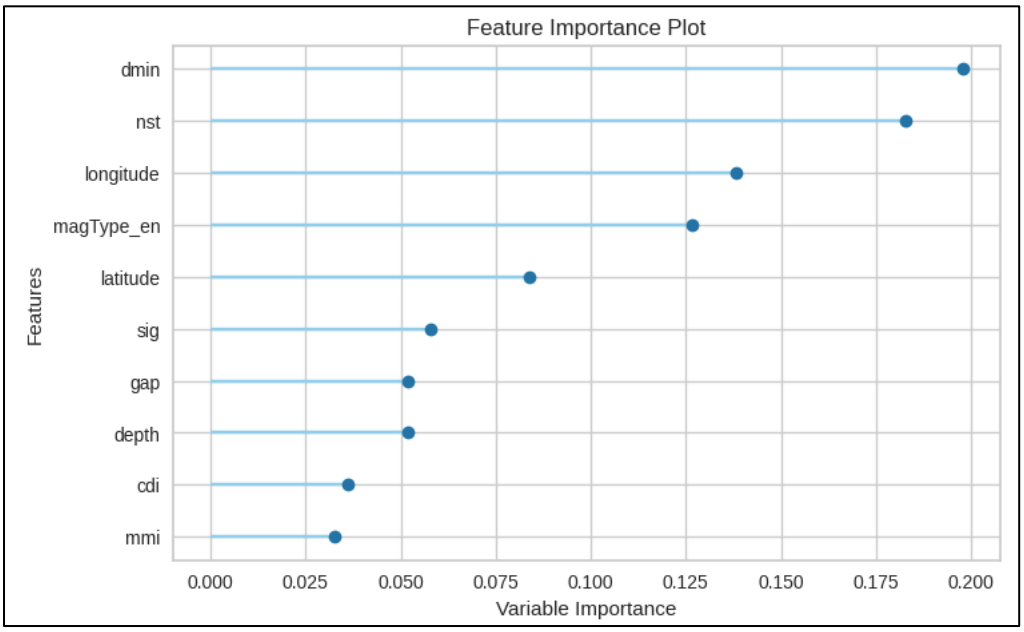
```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='sqrt',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_samples_leaf=1,
                       min_samples_split=2, min_weight_fraction_leaf=0.0,
                       monotonic_cst=None, n_estimators=100, n_jobs=-1,
                       oob_score=False, random_state=123, verbose=0,
                       warm_start=False)
```

(ค่าพารามิเตอร์ที่มีประสิทธิภาพมากที่สุดสำหรับโมเดล Random Forest Classifier (rf))

- กราฟ ROC Curve (Receiver Operating Characteristic Curve) ของ Random Forest Classifier (rf)



- การพล็อตกราฟที่เกี่ยวข้องกับคุณลักษณะ (features) หรือตัวแปรที่มีผลต่อการตัดสินใจของโมเดล Random Forest Classifier (rf)



จากภาพ แสดง features ที่มีผลต่อการตัดสินใจของโมเดล Random Forest ใน 5 ลำดับแรก มีดังนี้ dmin, nst, magType_en , longitude และ magnitude

- Generate test design : เป็นขั้นตอนที่เกี่ยวข้องกับการสร้างและทดสอบโมเดล โดยแบ่งชุดข้อมูลอินพุทเป็นชุดข้อมูลการฝึก (Train Set) และชุดข้อมูลทดสอบ (Test Set) เพื่อใช้ในการสร้างและประเมินประสิทธิภาพของโมเดลต่อไป

- โมเดล XGBoost :

1. Split Data into Train and Test Sets: ขั้นตอนนี้ใช้ฟังก์ชัน train_test_split จากไลบรารี Scikit-learn เพื่อแบ่งข้อมูลอินพุทและเป้าหมายเป็นชุดข้อมูลการฝึกและชุดข้อมูลทดสอบ โดยการกำหนดอัตราส่วนของชุดข้อมูลทดสอบ (test_size) เพื่อระบุส่วนส่วนของข้อมูลที่จะถูกใช้สำหรับทดสอบโมเดล โดยใช้อัตราส่วนประมาณ 70% ของข้อมูลสำหรับการฝึกและ 30% สำหรับการทดสอบ

2. สร้างโมเดล XGBClassifier ด้วยพารามิเตอร์ที่ถูกปรับค่า

3. เทรนโมเดล XGBClassifier ด้วยชุดข้อมูลการฝึกที่เตรียมไว้ โดยใช้ฟังก์ชัน fit เพื่อปรับโมเดลให้เหมาะสมกับข้อมูลการฝึกและเรียนรู้ความสัมพันธ์ระหว่างตัวแปรอินพุทและเป้าหมายที่ต้องการทำนาย

- โมเดล Random Forest Classifier

1. Split Data into Train and Test Sets: ขั้นตอนนี้ใช้ฟังก์ชัน train_test_split จากไลบรารี Scikit-learn เพื่อแบ่งข้อมูลอินพุทและเป้าหมายเป็นชุดข้อมูลการฝึกและชุดข้อมูลทดสอบ โดยการกำหนดอัตราส่วนของชุดข้อมูล

ทดสอบ (test_size) เพื่อระบุส่วนส่วนของข้อมูลที่จะถูกใช้สำหรับทดสอบโมเดล โดยใช้อัตราส่วนประมาณ 70% ของข้อมูลสำหรับการฝึกและ 30% สำหรับการทดสอบ

2. สร้างโมเดล RandomForestClassifier ด้วยพารามิเตอร์ที่ถูกปรับค่า

3. เทรนโมเดล RandomForestClassifier ด้วยชุดข้อมูลการฝึกที่เตรียมไว้ โดยใช้ฟังก์ชัน fit เพื่อปรับโมเดลให้เหมาะสมกับข้อมูลการฝึกและเรียนรู้ความสัมพันธ์ระหว่างตัวแปรอินพุทและเป้าหมายที่ต้องการทำนาย

- Build model : การสร้างโมเดลเพื่อทำนายผลลัพธ์จากข้อมูลที่มีอยู่ โดยเริ่มจาก

  - XGBoost : XGBoost ย่อมาจาก "eXtreme Gradient Boosting" เป็นอัลกอริทึม Machine Learning ประเภท Gradient Boosting ที่ใช้ Decision Tree เป็นโมเดลพื้นฐาน XGBoost

  โดยนำเอา Decision Tree มา train ต่อๆกันหลายๆ tree โดยที่แต่ละ decision tree จะเรียนรู้จาก error ของ tree ก่อนหน้า ทำให้ความแม่นยำของในการทำ prediction จะแม่นยำมากขึ้นเรื่อยๆ เมื่อมีการเรียนรู้ของ tree ต่อเนื่องกันจนมีความลึกมากพอ และ model จะหยุดเรียนรู้เมื่อไม่เหลือ pattern ของ error จาก tree ก่อนหน้าให้เรียนรู้แล้ว

  <u>การทำงานของ XGBoost</u> ดังนี้

  1. การสร้างต้นไม้เดี่ยว (Individual Tree Construction)
     - XGBoost จะสร้างต้นไม้ตัดสินใจแบบลำดับขั้น (Decision Tree) ซึ่งเป็นแบบจำลองพื้นฐาน
     - ต้นไม้แรกจะสร้างจากข้อมูลฝึกสอนทั้งหมด
  2. การรวมต้นไม้หลายๆต้น (Tree Ensemble)
     - หลังจากสร้างต้นไม้แรก XGBoost จะพิจารณาผลต่างระหว่างค่าจริงและค่าที่ได้จากแบบจำลอง (Residual)
     - จากนั้นสร้างต้นไม้ที่สองด้วยการพยายามลดผลต่างหรือ Residual นั้นลง
     - กระบวนการจะดำเนินต่อไปโดยสร้างต้นไม้ใหม่เพื่อลดผลต่างจากแบบจำลองก่อนหน้าลงเรื่อยๆ
  3. การปรับแบบจำลอง (Model Tuning)
     - XGBoost มีพารามิเตอร์หลายตัวที่สามารถปรับแต่งได้ เช่น จำนวนต้นไม้สูงสุด ลึกของต้นไม้ อัตราการเรียนรู้ เป็นต้น ซึ่งการปรับพารามิเตอร์เหล่านี้ให้เหมาะสมจะช่วยเพิ่มประสิทธิภาพของแบบจำลอง
  4. วนซ้ำจนกว่าจะบรรลุเงื่อนไขการหยุด เช่น จำนวน iteration สูงสุด หรือ error ต่ำสุด

  <u>ทำนายบนชุดข้อมูลทดสอบ</u> : ขั้นตอนที่ใช้โมเดลที่ถูกปรับค่าจาก (tuned_ xgboost_model) ในการทำนายผลลัพธ์ (y_pred) จากชุดข้อมูลทดสอบ (X_test) ซึ่งเป็นข้อมูลที่โมเดลไม่เคยเห็นมาก่อน เมื่อได้ผลลัพธ์การทำนายแล้ว เราสามารถนำไปใช้ในการประเมินประสิทธิภาพของโมเดลได้

- RandomForestClassifier : เป็นอัลกอริธึมการเรียนรู้แบบมีการควบคุม (Supervised Learning) ที่ใช้สำหรับปัญหาการจำแนกประเภท (Classification) โดยอยู่ภายใต้กลุ่มของ Ensemble Learning คือ การรวมตัวของหลายๆ แบบจำลองเข้าด้วยกัน

การทำงานของ RandomForestClassifier มีดังนี้

1. สร้างป่าของต้นไม้ตัดสินใจ (Decision Trees)

- RandomForestClassifier จะสร้างต้นไม้ตัดสินใจจำนวนมากๆ (เช่น 100-500 ต้น) จากชุดข้อมูลฝึกสอน

- ในการสร้างต้นไม้แต่ละต้น จะใช้เทคนิค Bagging (Bootstrap Aggregation) คือสุ่มเลือกข้อมูลจำนวนหนึ่งมาสร้างต้นไม้ โดยแต่ละต้นจะใช้ชุดข้อมูลที่แตกต่างกัน

- นอกจากนี้ ในการแบ่งโหนดของแต่ละต้น จะสุ่มเลือกคุณลักษณะ (Features) จำนวนจำกัดมาใช้เป็นเกณฑ์ในการแบ่งโหนด

2. ทำนายผลลัพธ์

- เมื่อต้องการทำนายผลของตัวอย่างใหม่ จะนำตัวอย่างนั้นผ่านไปยังต้นไม้ทุกๆ ต้นในป่า แต่ละต้นจะทำนายผลลัพธ์ของตัวอย่างนั้น

- RandomForestClassifier จะรวบรวมผลลัพธ์จากทุกๆ ต้น และเลือกประเภทที่ได้รับคะแนนเสียงข้างมากเป็นผลลัพธ์สุดท้าย

3. การหาค่าพารามิเตอร์ที่เหมาะสม

- RandomForestClassifier มีพารามิเตอร์สำคัญหลายตัวที่ต้องปรับให้เหมาะสม เช่น จำนวนต้นไม้ ลึกของต้นไม้ จำนวนคุณลักษณะที่ใช้ในการแบ่งโหนด เป็นต้น

ทำนายบนชุดข้อมูลทดสอบ : ขั้นตอนที่ใช้โมเดลที่ถูกปรับค่าจาก (tuned_rf_model) ในการทำนายผลลัพธ์ (y_pred) จากชุดข้อมูลทดสอบ (X_test) ซึ่งเป็นข้อมูลที่โมเดลไม่เคยเห็นมาก่อน เมื่อได้ผลลัพธ์การทำนายแล้ว เราสามารถนำไปใช้ในการประเมินประสิทธิภาพของโมเดลได้

5. Evaluation : ขั้นตอนนี้เป็นการ import functions ที่ใช้ในการคำนวณค่าประเมินความสามารถของโมเดลที่ได้ตามต้องการ เช่น precision, accuracy, recall, f1 score และ confusion matrix จากไลบรารี sklearn.metrics

5.1 จากการคำนวณค่าประเมินโมเดลของ xgboost ได้ผลลัพธ์ดังนี้

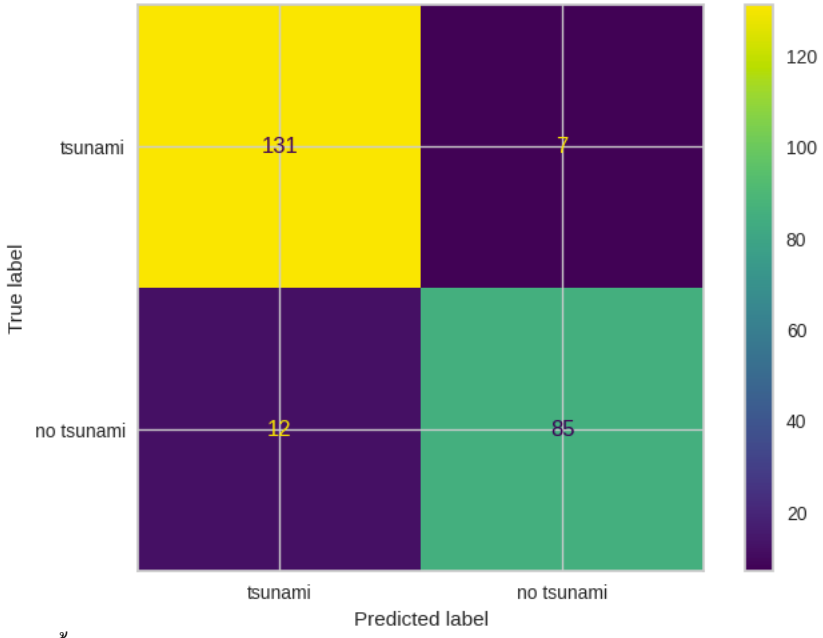
Accuracy: 0.9191489361702128

Precision: 0.9239130434782609

Recall: 0.8762886597938144

F1-Score: 0.8994708994708994

จากนั้น พล็อต Confusion Matrix โดยใช้ ConfusionMatrixDisplay เพื่อสร้างวิธีการแสดงผลของ Confusion Matrix และเป็นการระบุ confusion matrix ที่เกิดขึ้นจากการทำนายโมเดล XGBoost ซึ่งจะแสดงในรูปแบบของกริดสี่เหลี่ยมจัตุรัสที่แสดงจำนวนของความผิดพลาดที่เกิดขึ้นในการทำนายของโมเดล โดยแกน x และแกน y จะแสดงคลาสที่ทำนายและคลาสที่เป็นจริง ซึ่งในกรณีนี้คือ 'tsunami' และ 'no tsunami' ตามลำดับ



สามารถแปลผลได้ดังนี้

จำนวน 131 คือ จำนวนที่เป็น Tsunami ทำนายถูกต้องว่าเป็น Tsunami

จำนวน 7 คือ จำนวนที่เป็น Tsunami แต่ทำนายผิดพลาดว่าเป็น no tsunami

จำนวน 12 คือ จำนวนที่เป็น no tsunami แต่ทำนายผิดพลาดว่าเป็น Tsunami

จำนวน 85 คือ จำนวนที่เป็น no tsunami ทำนายถูกต้องว่าเป็น no tsunami

5.2 จากการคำนวณค่าประเมินโมเดลของ Random Forest Classifier ได้ผลลัพธ์ดังนี้

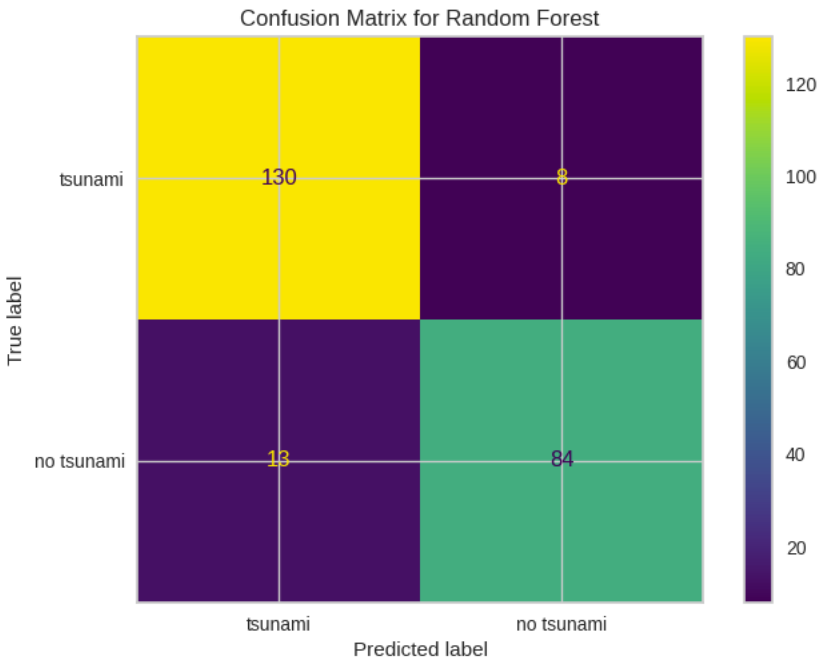
Accuracy: 0.9106382978723404

Precision: 0.9130434782608695

Recall: 0.865979381443299

F1-Score: 0.8888888888888888

จากนั้น พล็อต Confusion Matrix โดยใช้ ConfusionMatrixDisplay เพื่อสร้างวิธีการแสดงผลของ Confusion Matrix และเป็นการระบุ confusion matrix ที่เกิดขึ้นจากการทำนายโมเดล Random Forest Classifier ซึ่งจะแสดงในรูปแบบของกริดสี่เหลี่ยมจัตุรัสที่แสดงจำนวนของความผิดพลาดที่เกิดขึ้นในการทำนายของโมเดล โดยแกน x และแกน y จะแสดงคลาสที่ทำนายและคลาสที่เป็นจริง ซึ่งในกรณีนี้คือ 'tsunami' และ 'no tsunami' ตามลำดับ



สามารถแปลผลได้ดังนี้

จำนวน 130 คือ จำนวนที่เป็น Tsunami ทำนายถูกต้องว่าเป็น Tsunami

จำนวน 8 คือ จำนวนที่เป็น Tsunami แต่ทำนายผิดพลาดว่าเป็น no tsunami

จำนวน 13 คือ จำนวนที่เป็น no tsunami แต่ทำนายผิดพลาดว่าเป็น Tsunami

จำนวน 84 คือ จำนวนที่เป็น no tsunami ทำนายถูกต้องว่าเป็น no tsunami

จากขั้นตอน Evaluation ได้ผลลัพธ์จากการคำนวณค่าประเมินโมเดลของ XGBoost และ โมเดล Random Forest Classifier นั้น จะเห็นได้ว่า XGBoost มีค่าประเมินที่ดีกว่าในเกือบทุกการประเมินดังนี้

ดังนี้

- Accuracy: XGBoost (0.9191) > Random Forest (0.9106)
- Precision: XGBoost (0.9239) > Random Forest (0.9130)

- Recall: XGBoost (0.8763) > Random Forest (0.8660)
- F1-Score: XGBoost (0.8995) > Random Forest (0.8889)

ดังนั้น หากพิจารณาจากค่าประเมินโมเดลเพียงอย่างเดียว แนะนำว่าควรใช้โมเดล XGBoost เนื่องจากมีประสิทธิภาพที่ดีกว่า Random Forest ทั้งในเรื่องความถูกต้อง ค่าความแม่นยำ ค่าความครบถ้วน และค่า F1-Score

6. Deploy : เมื่อนำโมเดลที่สร้างขึ้นเพื่อทำนายการเกิดสึนามิไปใช้งานจริง มีข้อแนะนำสำคัญดังนี้

- ควรมีระบบตรวจสอบและยืนยันผลการทำนาย และไม่ควรนำไปใช้งานโดยตรง แต่ควรมีการตรวจสอบและยืนยันความถูกต้องอีกครั้งโดยผู้เชี่ยวชาญ

- จัดทำระบบการแจ้งเตือนและการสื่อสารผลการทำนายอย่างมีประสิทธิภาพ

- เนื่องจากสภาพแวดล้อมและปัจจัยที่เกี่ยวข้องกับการเกิดสึนามิมีการเปลี่ยนแปลงอยู่เสมอ โมเดลจำเป็นต้องได้รับการปรับปรุงและอัพเดตอย่างสม่ำเสมอ

- จำเป็นต้องมีแผนรองรับเหตุการณ์ฉุกเฉิน เช่น การแจ้งเตือนประชาชน การอพยพ การให้ความช่วยเหลือ เพื่อลดความเสียหายให้มากที่สุด

Code : https://colab.research.google.com/drive/1pWTqIGxf5YtW3tQIgvowxIpqn49J2Soj?usp=sharing