TOPIK KHUSUS BIG DATA

Anomaly Detection on Indonesian Earthquake Data

Disusun Oleh:

Imran Y. A. Abu libda D121211105

TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

2024

Anomaly Detection on Indonesian Earthquake Data

Introduction

This report is based on the application of techniques on anomaly detection using machine learning on the dataset representing earthquake activities. The aim is to point to some sort of anomaly in the given dataset, which should have been indicative of some outliers or unusual events, such as earthquakes that might have resulted in tsunamis.

At the end, this detected anomaly was verified by comparing with the actual tsunami occurrences in order to measure the accuracy level of the models used.

Dataset

The earthquake dataset used historical earthquake data of 16 years from BMKG (https://repogempa.bmkg.go.id/eventcatalog), which includes the following features:

- tgl: The date of the earthquake.
- ot: The time of the earthquake.
- lat: The latitude where the earthquake occurred.
- lon: The longitude where the earthquake occurred.
- depth: The depth of the earthquake (in kilometers).
- mag: The magnitude of the earthquake.
- remark: A textual description of the location of the earthquake.

5 rows 5 rows × 7 columns

	tgl	ot	lat	lon	depth	mag	remark
0	2008/11/01	21:02:43.058	-9.18	119.06	10	4.9	Sumba Region - Indonesia
1	2008/11/01	20:58:50.248	-6.55	129.64	10	4.6	Banda Sea
2	2008/11/01	17:43:12.941	-7.01	106.63	121	3.7	Java - Indonesia
3	2008/11/01	16:24:14.755	-3.30	127.85	10	3.2	Seram - Indonesia
4	2008/11/01	16:20:37.327	-6.41	129.54	70	4.3	Banda Sea

We also included additional features for better temporal analysis:

- datetime: A timestamp combining the date (tgl) and time (ot) of the earthquake.
- year, month, day, hour, minute: Separate columns extracted from the datetime field for year, month, day, hour, and minute.

Also I used a tsunami dataset, which contains information on tsunamis, including the year, month, day, and time of each tsunami event, for comparison with detected anomalies.

```
# Convert 'tgl' and 'ot' to datetime
df['datetime'] = pd.to_datetime(df['tgl'] + ' ' + df['ot'])
[51]

# Extract year, month, day, and hour for analysis
df['year'] = df['datetime'].dt.year
df['month'] = df['datetime'].dt.month
df['day'] = df['datetime'].dt.day
df['hour'] = df['datetime'].dt.hour
df['minute'] = df['datetime'].dt.minute

# Display the first few rows of the dataset
df.head()
[52]
```

5 rows | 5 rows × 13 columns | Static Output

	tgl	ot	lat	lon	depth	mag	remark	datetime	year	month
0	2008/11/01	21:02:43.058	-9.18	119.06	10	4.9	Sumba Region - Indonesia	2008-11-01 21:02:43.058	2008	11
1	2008/11/01	20:58:50.248	-6.55	129.64	10	4.6	Banda Sea	2008-11-01 20:58:50.248	2008	11
2	2008/11/01	17:43:12.941	-7.01	106.63	121	3.7	Java - Indonesia	2008-11-01 17:43:12.941	2008	11
3	2008/11/01	16:24:14.755	-3.30	127.85	10	3.2	Seram - Indonesia	2008-11-01 16:24:14.755	2008	11
4	2008/11/01	16:20:37.327	-6.41	129.54	70	4.3	Banda Sea	2008-11-01 16:20:37.327	2008	11

Methodology

1. Data Preprocessing

- **Merging Date and Time**: I combined the tgl and ot columns to create a single datetime column to simplify temporal analysis.
- **Feature Extraction**: Extracted additional features such as year, month, day, hour, and minute from the datetime field.

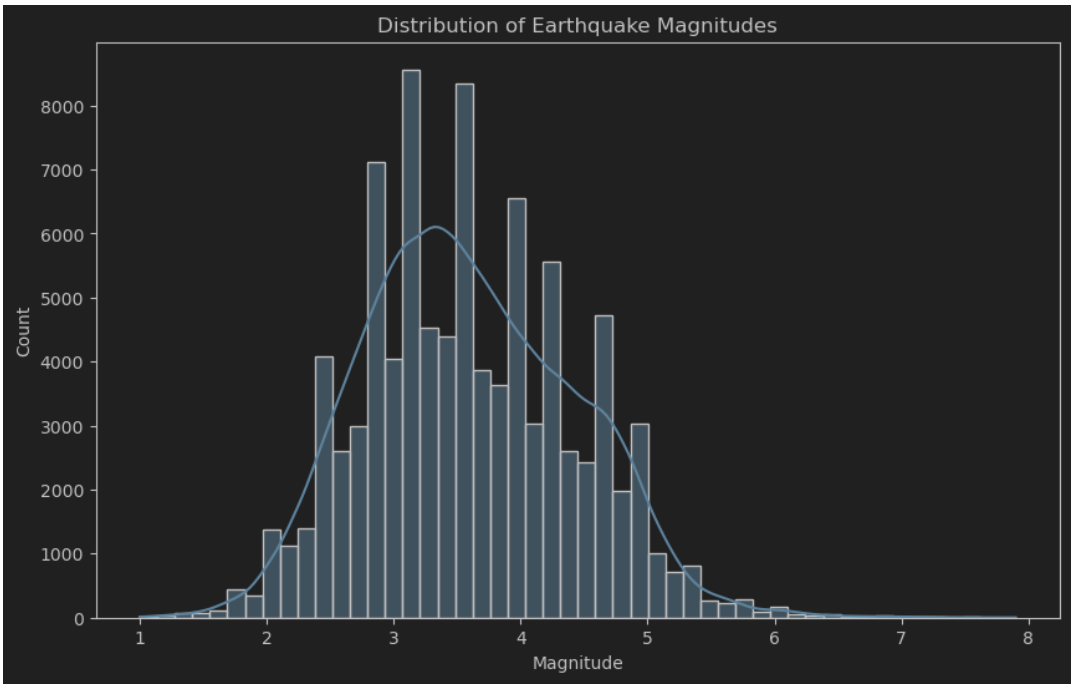
```
# Check for missing values
missing_values = df.isnull().sum()

# Display percentage of missing values for each column
missing_percentage = (missing_values / len(df)) * 100
print(missing_percentage)

# Check for duplicates
duplicate_count = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicate_count}")

[46]
```

```
tgl          0.000000
ot           0.000000
lat          0.000000
lon          0.000000
```

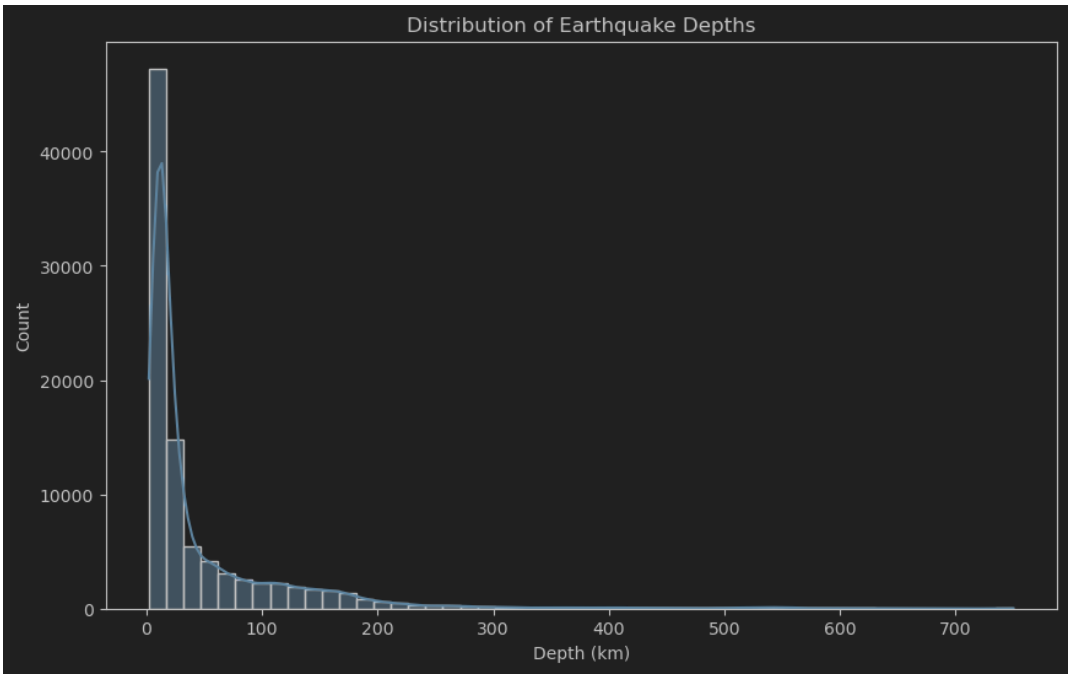
```
# Drop columns with more than 50% missing values
df = df.dropna(thresh=0.5*len(df), axis=1)
#df = df.dropna('agency')
[47]
```

2. Exploratory Data Analysis (EDA)

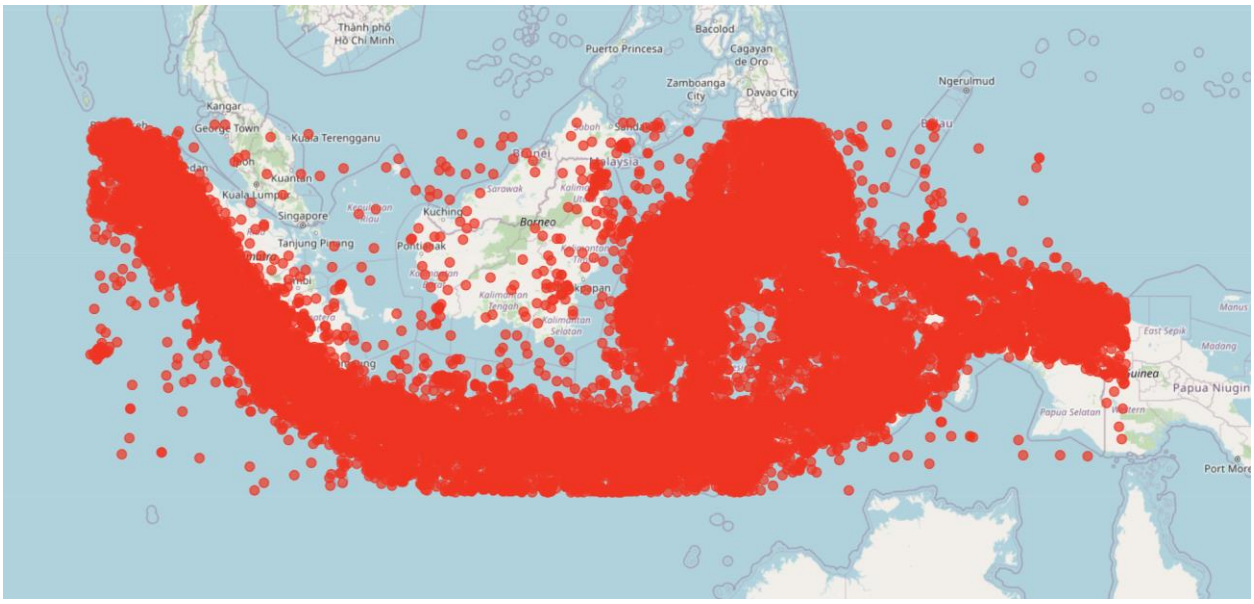
- **Distribution of Magnitude**: I analyzed the distribution of the magnitude of earthquakes to identify common ranges and potential outliers.

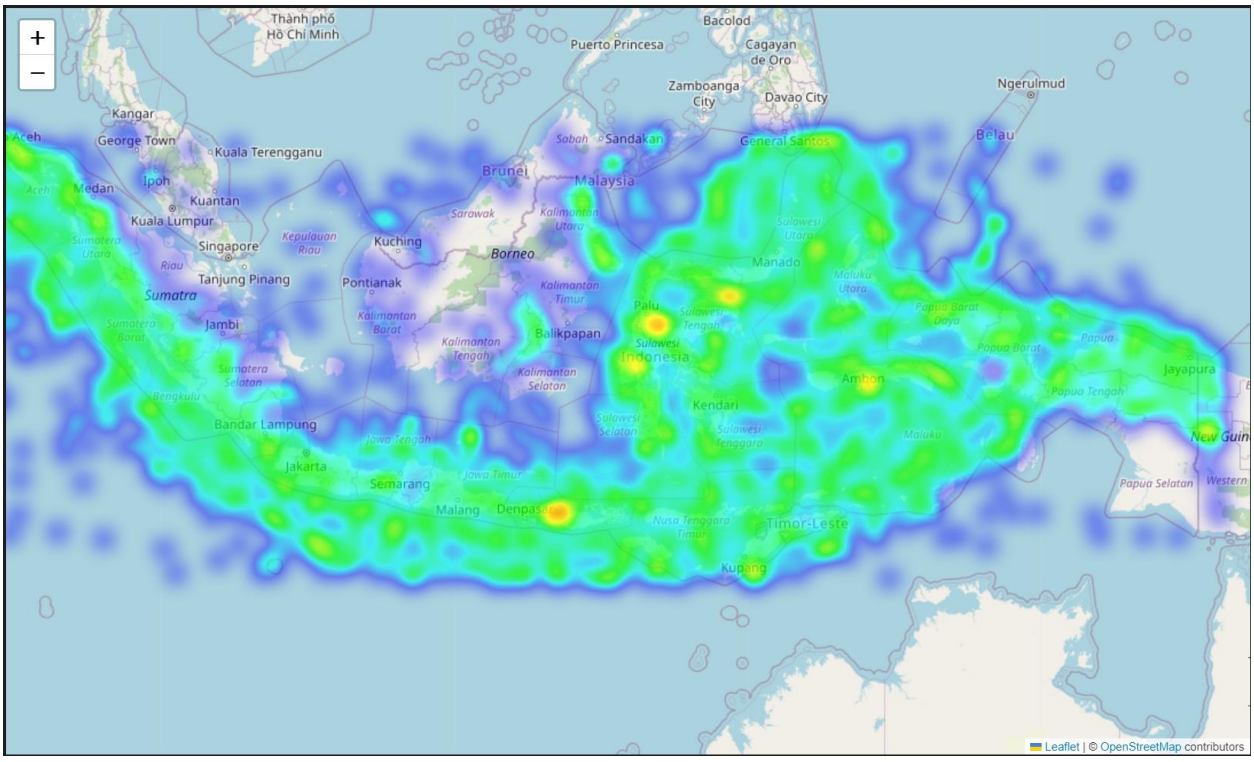


- **Depth Distribution**: We examined the depth of the earthquakes to understand how deep earthquakes vary and whether depth plays a role in anomalous events.



- **Geographical Visualization**: We plotted the earthquakes on a map to analyze their geographical distribution and determine whether anomalies tend to cluster in specific regions.





3. Anomaly Detection Techniques

I used three machine learning techniques to detect anomalies in the dataset:

- **Isolation Forest**: This model isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

```
features = ['lat', 'lon', 'depth', 'mag']  # Feature columns

# Initialize Isolation Forest
iso_forest = IsolationForest(contamination=0.05, random_state=42)

# Fit the model to the data
iso_forest.fit(df[features])

# Predict anomalies
df['anomaly_iso'] = iso_forest.predict(df[features])

# Map the output to more readable format
df['anomaly_iso'] = df['anomaly_iso'].map({1: 'Normal', -1: 'Anomaly'})

# Count the number of anomalies detected
anomalies_iso = df[df['anomaly_iso'] == 'Anomaly']
print(f'Number of anomalies detected by Isolation Forest: {len(anomalies_iso)}')

# Display the anomalies
print(anomalies_iso[['lat', 'lon', 'depth', 'mag', 'remark', 'datetime']])

✓ [2] 929ms
```

```
                    datetime
21     2008-11-02 21:59:16.852
26     2008-11-02 00:10:37.651
35     2008-11-03 19:22:03.486
64     2008-11-07 22:36:26.841
73     2008-11-07 11:32:44.802
...                        ...
92723  2023-01-23 07:07:26.782
92749  2023-01-24 18:09:10.201
92836  2023-01-26 21:54:20.115
92858  2023-01-26 08:15:35.234
92876  2023-01-26 03:33:28.683

[4645 rows x 6 columns]
```

- **One-Class SVM**: One-Class SVM finds a decision function that best separates the data points from the origin in a high-dimensional space, identifying the most unusual data points as anomalies.

```python
# Standardize the data to ensure better performance of the SVM
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df[features])

# Initialize One-Class SVM
svm_model = OneClassSVM(kernel='rbf', nu=0.05, gamma='auto')

# Fit the model to the scaled data
svm_model.fit(scaled_features)

# Predict anomalies
df['anomaly_svm'] = svm_model.predict(scaled_features)

# Map the output to more readable format
df['anomaly_svm'] = df['anomaly_svm'].map({1: 'Normal', -1: 'Anomaly'})

# Count the number of anomalies detected
anomalies_svm = df[df['anomaly_svm'] == 'Anomaly']
print(f'Number of anomalies detected by One-Class SVM: {len(anomalies_svm)}')

# Display the anomalies
print(anomalies_svm[['lat', 'lon', 'depth', 'mag', 'remark', 'datetime']])
```

✓ [3] 56s 697ms

```
                    datetime
21     2008-11-02 21:59:16.852
26     2008-11-02 00:10:37.651
70     2008-11-07 16:04:27.451
87     2008-11-08 04:00:18.077
104    2008-11-10 07:59:48.643
...                        ...
92808  2023-01-25 17:58:05.278
92836  2023-01-26 21:54:20.115
92872  2023-01-26 04:23:50.693
92873  2023-01-26 04:23:50.693
92881  2023-01-26 02:41:07.016

[4651 rows x 6 columns]
```

- **Elliptic Envelope**: This model fits a multivariate Gaussian distribution to the data and determines the points that do not fit within this distribution as anomalies.

```
elliptic_env.fit(X)
```

✓ [7] 29s 63ms

```
EllipticEnvelope(contamination=0.01)
```

```
# Predict anomalies using Elliptic Envelope
df['anomaly_elliptic'] = elliptic_env.predict(X)

# Convert predictions into a readable format
df['anomaly_elliptic'] = df['anomaly_elliptic'].map({1: 'Normal', -1: 'Anomaly'})

# Count the anomalies detected
anomalies_elliptic = df[df['anomaly_elliptic'] == 'Anomaly']
print(f'Total number of anomalies detected by Elliptic Envelope: {len(anomalies_elliptic)}')
```

✓ [9] 13ms

```
Total number of anomalies detected by Elliptic Envelope: 929
```

```
# Display the anomalies detected by Elliptic Envelope
print(anomalies_elliptic[['lat', 'lon', 'depth', 'mag', 'remark', 'datetime']])
```

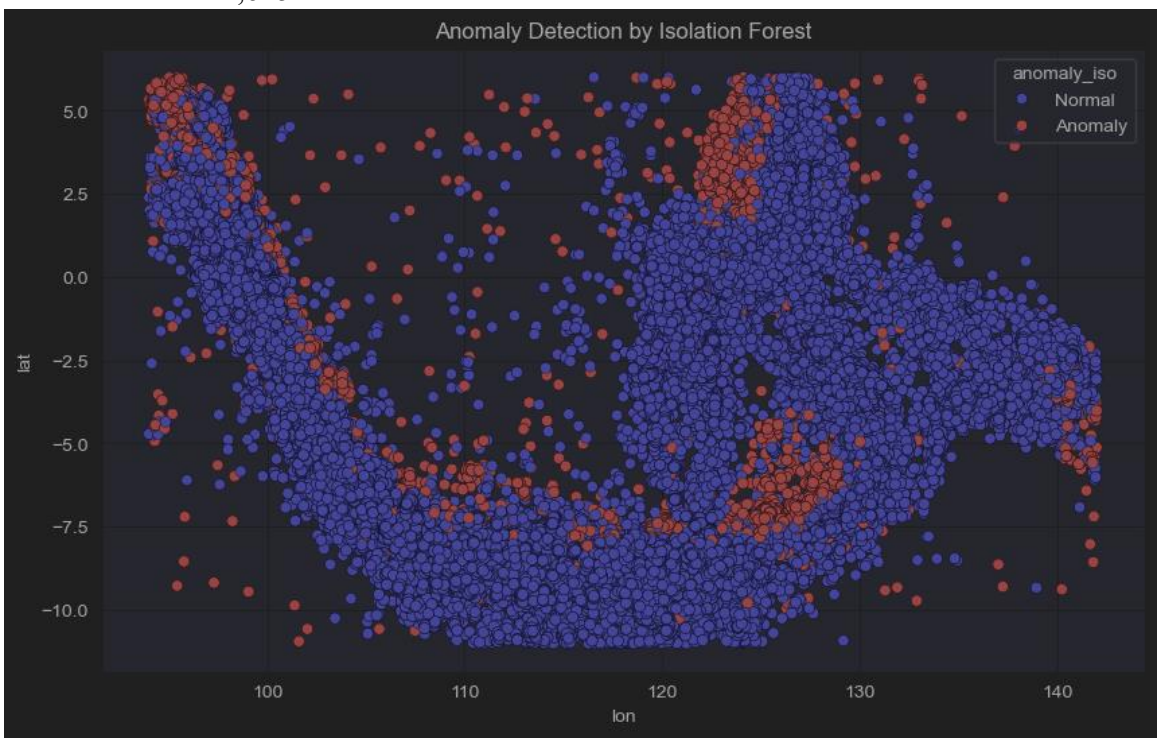
✓ [10] < 10 ms

```
         lat     lon  depth  mag               remark                datetime
21      3.06  121.83    588  4.4          Celebes Sea  2008-11-02 21:59:16.852
104    -7.25  121.00    650  4.5           Flores Sea  2008-11-10 07:59:48.643
146     2.29  124.73    650  5.1          Celebes Sea  2008-11-23 08:01:24.185
147    -5.00  115.99    551  4.9             Java Sea  2008-11-24 09:39:16.510
148     1.10  125.58    650  5.6  Northern Molucca Sea  2008-11-24 09:10:49.164
...      ...     ...    ...  ...                 ...                      ...
92042   4.02  122.90    554  5.1          Celebes Sea  2023-01-07 05:21:57.724
92109  -7.25  125.18    443  4.6            Banda Sea  2023-01-09 07:23:01.005
92537  -6.30  125.45    552  5.1            Banda Sea  2023-01-19 11:56:52.215
92607  -7.55  123.04    544  4.8            Banda Sea  2023-01-20 03:40:32.826
92720  -6.91  126.77    426  4.5            Banda Sea  2023-01-23 13:29:36.362

[929 rows x 6 columns]
```

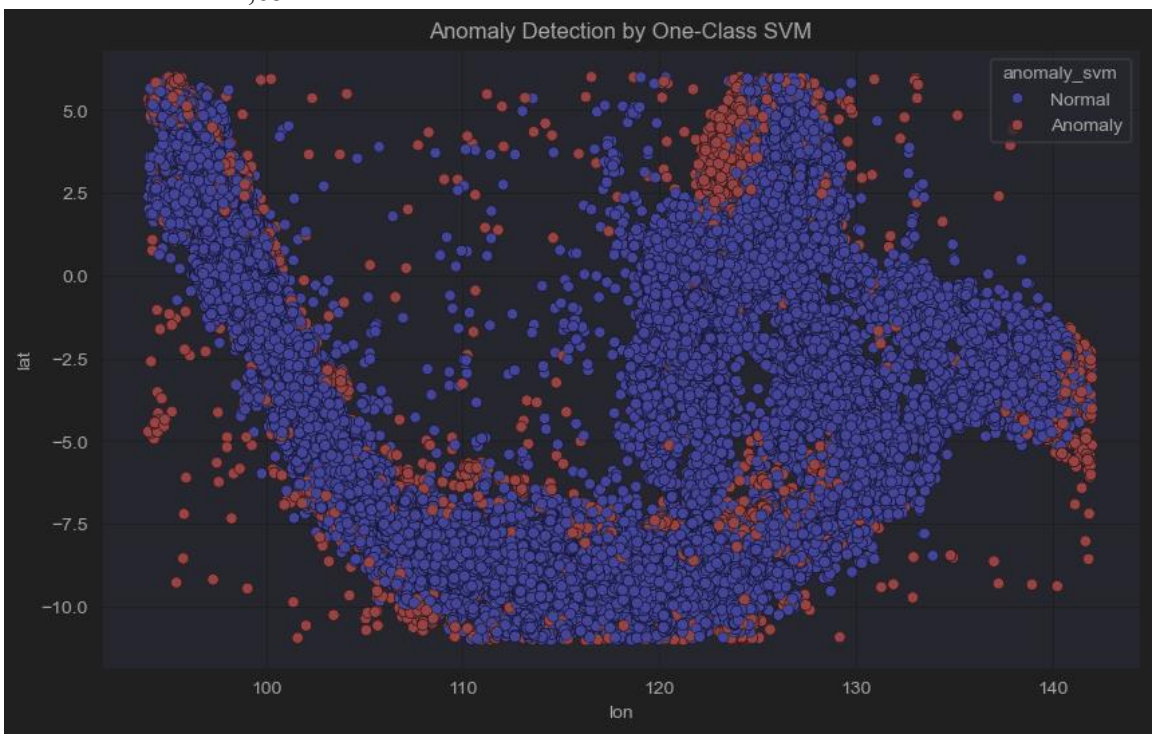
I tuned the contamination parameter (the proportion of the dataset to be considered as anomalies) to 0.01 for the Elliptic Envelope model, meaning that 1% of the data points were flagged as anomalies.

4. Model Comparison

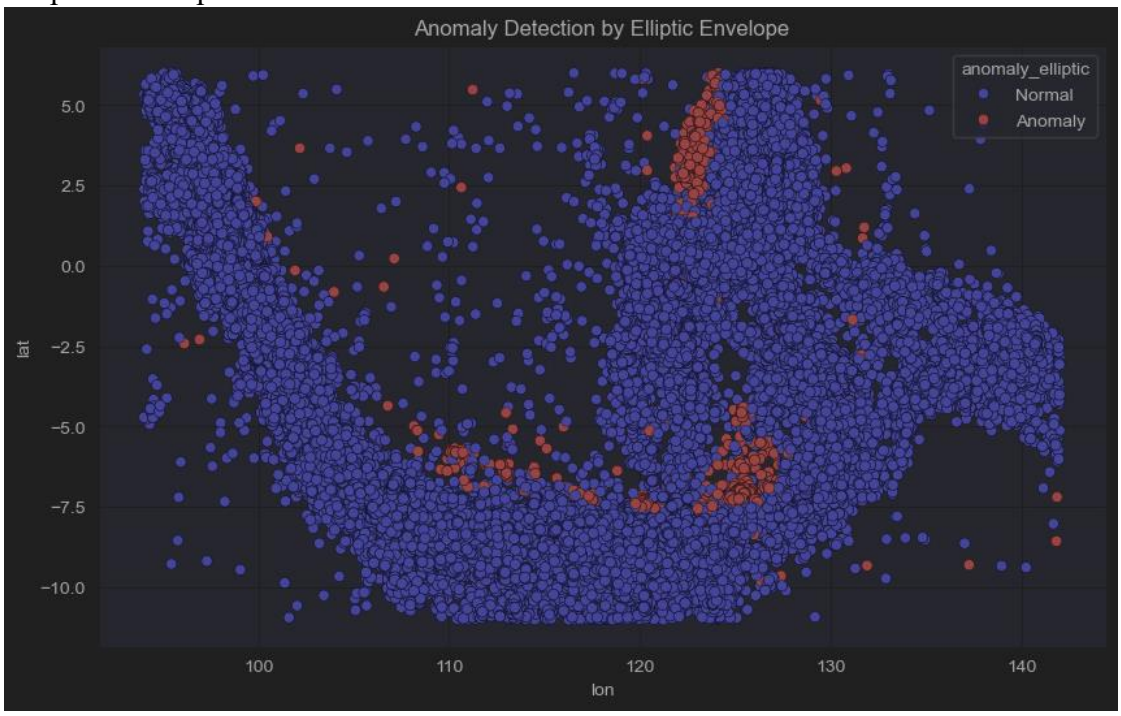
- **Number of Anomalies Detected**:
  - Isolation Forest: 4,645 anomalies



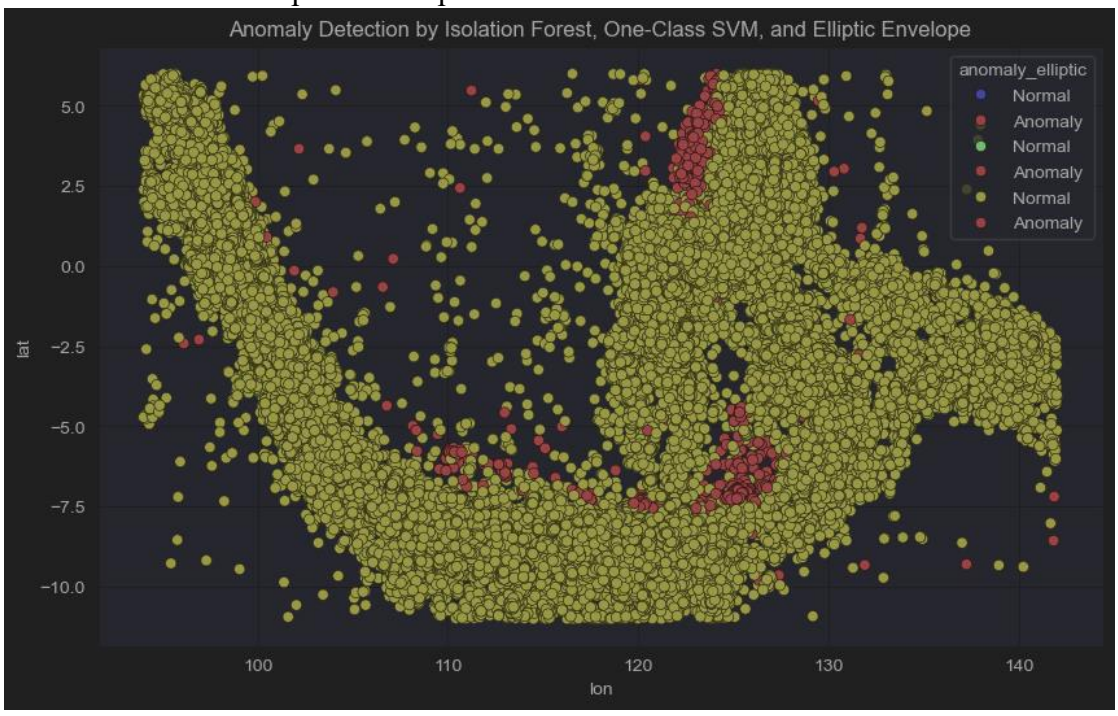
  - One-Class SVM: 4,651 anomalies



- Elliptic Envelope: 929 anomalies



- **Common Anomalies Between Models**:
  - Common between Elliptic Envelope and Isolation Forest: 1,061
  - Common between Elliptic Envelope and One-Class SVM: 861



The Elliptic Envelope model returned the fewest anomalies and could, therefore, be regarded as most conservative in anomaly detection. On the other hand, the anomalies it did capture had the largest similarities in facts among the other models.

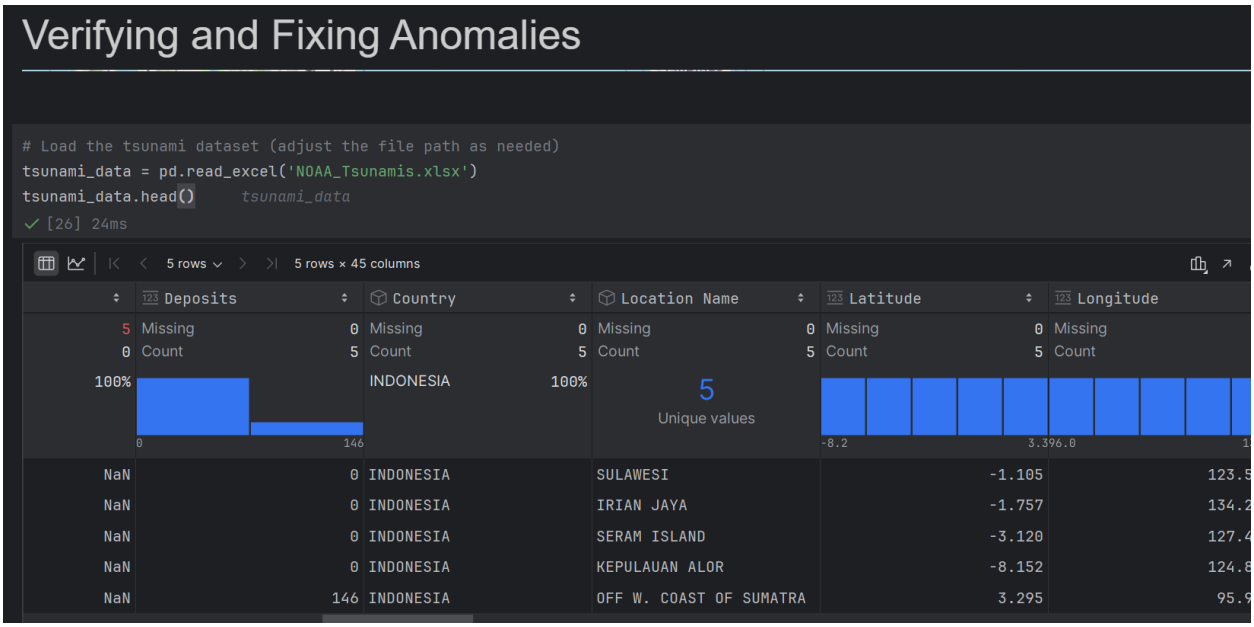
Considering the fact that the Elliptic Envelope model would take into consideration the Gaussian distribution, thus being more sensitive to smaller clusters of unusual data, I chose the Elliptic Envelope as the best performing model.

5. Tsunami Comparison

To validate the detected anomalies, we compared the anomalies from the Elliptic Envelope model with the actual tsunami occurrences from the tsunami dataset. The comparison was based on the year, month, and hour of the events to ensure that anomalies corresponded to actual tsunami events.

Methodology for Tsunami Comparison

- I loaded the tsunami dataset, which contains information about tsunamis, including the Year, Month, and Hour of each event.



- I merged the tsunami dataset with the anomalies detected by the Elliptic Envelope model, using the year, month, and hour columns as keys for comparison.

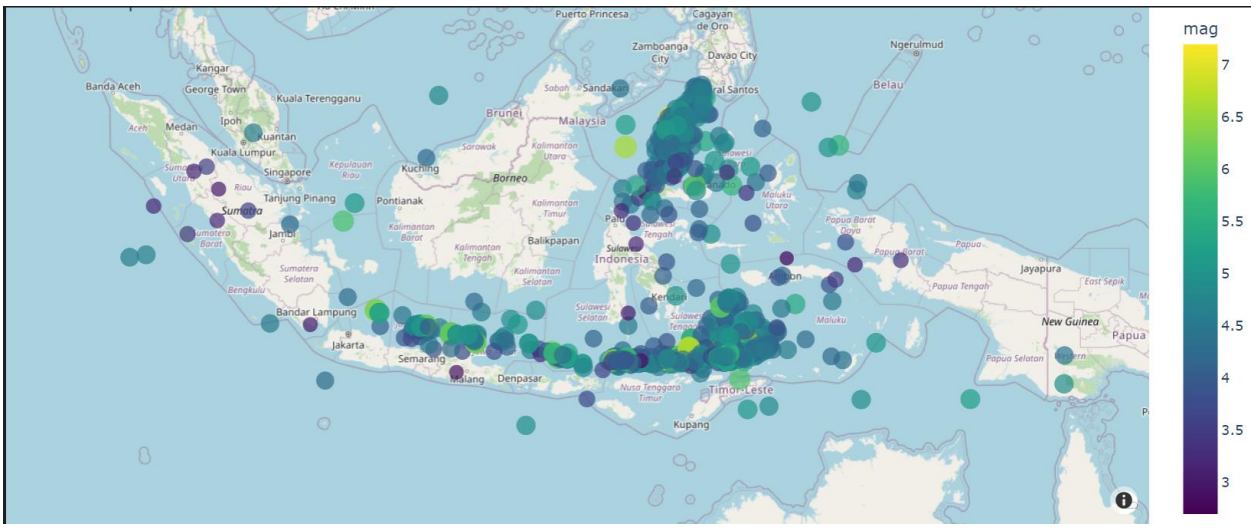
```
# Select only relevant columns from the tsunami dataset: Year, Mo (Month), Hr (Hour)
tsunami_data = tsunami_data[['Year', 'Mo', 'Dy','Hr']]

# Ensure both datasets have consistent column names and data types for comparison
anomalies['year'] = anomalies['year'].astype(int)
anomalies['month'] = anomalies['month'].astype(int)
anomalies['day'] = anomalies['day'].astype(int)
anomalies['hour'] = anomalies['hour'].astype(int)

# Merge datasets on year, month, and hour
common_anomalies_tsunami = pd.merge(anomalies, tsunami_data, left_on=['year', 'month', 'hour'], right_on=['Year', 'Mo', 'Hr'],
 how='inner')

# Display the common entries
print("Common entries between anomalies and tsunamis (based on year, month, and hour):")
print(common_anomalies_tsunami)
```

✓ [27] < 10 ms

```
Common entries between anomalies and tsunamis (based on year, month, and hour):
          tgl           ot   lat     lon  depth  mag  \
0  2009/09/30  07:27:39.049  2.58  122.94    472  4.8
1  2012/04/08  08:50:04.030 -6.92  125.29    539  4.3
2  2012/04/19  10:59:19.479  3.09  122.52    539  4.5
3  2018/07/05  22:37:04.395 -5.76  110.38    529  4.8
4  2018/07/05  22:37:04.395 -5.76  110.38    529  4.8
5  2018/08/13  04:04:26.407 -7.61  122.70    542  4.8
6  2019/08/10  12:59:16.432  0.30  120.25    539  4.4
7  2021/06/21  04:30:33.206  5.61  124.13    470  4.6
8  2021/12/23  03:01:31.794  5.68  123.95    497  4.6

                        remark               datetime  year  month  day  \
0                  Celebes Sea  2009-09-30 07:27:39.049  2009      9   30
```

Results of Tsunami Comparison

- After comparing the anomalies with the tsunami dataset, we found several common points where 9 anomalies matched the tsunami events based on the same year, month, and hour.



- This suggests that the Elliptic Envelope model was successful in identifying some events that correspond to actual tsunamis, although further investigation is necessary to verify the full scope of anomalies.

Recommendations

1. **Data Quality:**

   Government datasets are often inconsistent and/or partial. The data must be cleaned and preprocessed to remove any outliers and impute any missing values before any serious use.

   Improved consistency in recording earthquake events, such as consistently applying time zones, could improve the accuracy of the machine learning models.

2. **Better Feature Engineering:**

   Other features can be engineered to provide better accuracy of the model. For example, the inclusion of weather information or seismological features Fault lines can provide better accuracy of anomaly detection by the model.

3. **Using a Hybrid Approach:**

   A hybrid approach by combing a number of the anomaly detection techniques would more than likely yield better results. For example, an isolation forest or an elliptic envelope model alone and in use with each other will be employed.

4. **Smoothening the Contamination Rate:**

   The contamination rate, which is determining the share of anomalies to be returned, could be fine-tuned further using expert knowledge, as setting it too low may miss important anomalies, and too high may return too many false positives.

5. **Tsunami Early Warning System:**

   Anomalies detected by the machine learning model, such as the Elliptic Envelope, should be incorporated into an early warning system for tsunamis. In fact, such anomalies indicate unusual seismic activity and thus can provide an opportunity for the authorities to issue timely warnings that could save lives.

GitHub: **https://github.com/3m0r9/Earthquake-Anomaly-Detection-**