Data Science Project Number - 1

Data Science Project Number - 1

Name: Aarya Trifale, Roll no: 41

Domain: Underwater Acoustics

Problem Statement: Analyzing the Impact of Depth on Sound Intensity in Underwater Environments

[18]:

```
#Importing the required libraries and also importing our dataset
import pandas as pd
import numpy as np

file = pd.read_csv("underwater data for ds.csv")
file
```

[18]:

```
           sound_type       location                 time  intensity_db  \
0          Ship Noise   Indian Ocean  2023-01-01 00:00:00    125.126340
1    Seismic Activity  Pacific Ocean  2023-01-01 01:00:00    148.142563
2          Whale Call  Pacific Ocean  2023-01-01 02:00:00    134.256358
3          Ship Noise   Arctic Ocean  2023-01-01 03:00:00    111.346445
4          Ship Noise   Indian Ocean  2023-01-01 04:00:00    106.523780
..                …              …                    …             …
495        Ship Noise  Pacific Ocean  2023-01-21 15:00:00    112.036968
496        Ship Noise  Pacific Ocean  2023-01-21 16:00:00    112.658408
497  Seismic Activity   Arctic Ocean  2023-01-21 17:00:00    135.662413
498        Whale Call   Arctic Ocean  2023-01-21 18:00:00    130.228372
499  Seismic Activity   Indian Ocean  2023-01-21 19:00:00    147.700610

     depth_meters  duration_seconds  frequency_hz  water_temperature_c  \
0      558.392819             200.0   7503.311575            20.944851
1      464.070791              79.0  19015.271842            16.082891
2      559.065483             173.0  14645.238957             9.285828
3      610.870358              69.0  11981.196514            24.413851
4      582.048218              87.0   3137.252436            20.541935
..              …                 …             …                    …
495    451.208859             251.0   7079.977516             2.747462
496    715.730821             124.0  11681.449115            27.519407
497    439.428508             121.0   1573.138047             4.104559
498    574.209537               NaN  19488.408257            28.507121
```

```
499   529.929258             120.0          NaN            13.380173

     salinity_psu  distance_from_source_m  ambient_noise_db
0             NaN             5238.909673               NaN
1       33.793307             4843.900589               NaN
2       36.110621              353.856451               NaN
3       35.125574             3478.353493         57.486386
4       35.645928             3863.936626         58.158492
..            …                       …               …
495     34.677490             6603.856047         64.769764
496     34.336432             9570.484749         67.318371
497     33.244458              782.684362         75.967314
498     32.658500              664.841739         79.422180
499     36.043336             2893.652039         62.227526

[500 rows x 11 columns]
```

Descriptions for each column in the database:

- **sound_type**: The type of sound detected (e.g., Ship Noise, Seismic Activity, Whale Call).
- **location**: The geographical location where the sound was recorded (e.g., Indian Ocean, Pacific Ocean).
- **time**: The timestamp of the sound event, indicating when it was recorded.
- **intensity_db**: The intensity of the sound in decibels (dB).
- **depth_meters**: The depth (in meters) at which the sound was detected.
- **duration_seconds**: The duration of the sound event, measured in seconds.
- **frequency_hz**: The estimated frequency of the sound (in Hz).
- **water_temperature_c**: The water temperature (in degrees Celsius) at the depth of the sound detection.
- **salinity_psu**: The salinity of the water in Practical Salinity Units (PSU).
- **distance_from_source_m**: The estimated distance (in meters) between the recording device and the sound source.
- **ambient_noise_db**: The level of ambient noise (in decibels) present during the recording.

[19]:
```
file.head() #.head() returns the first few rows (the "head" of the dataframe i.
↪e. excel file)
```

[19]:
```
         sound_type       location                 time  intensity_db  \
0        Ship Noise   Indian Ocean  2023-01-01 00:00:00    125.126340
1  Seismic Activity  Pacific Ocean  2023-01-01 01:00:00    148.142563
2        Whale Call  Pacific Ocean  2023-01-01 02:00:00    134.256358
3        Ship Noise   Arctic Ocean  2023-01-01 03:00:00    111.346445
4        Ship Noise   Indian Ocean  2023-01-01 04:00:00    106.523780

   depth_meters  duration_seconds  frequency_hz  water_temperature_c  \
0    558.392819             200.0   7503.311575            20.944851
1    464.070791              79.0  19015.271842            16.082891
2    559.065483             173.0  14645.238957             9.285828
3    610.870358              69.0  11981.196514            24.413851
```

```
4    582.048218              87.0   3137.252436          20.541935

   salinity_psu  distance_from_source_m  ambient_noise_db
0           NaN             5238.909673               NaN
1     33.793307             4843.900589               NaN
2     36.110621              353.856451               NaN
3     35.125574             3478.353493         57.486386
4     35.645928             3863.936626         58.158492
```

[20]:
```
file.tail() #.tail() method returns a specified number of last row
```

[20]:
```
           sound_type       location                 time  intensity_db  \
495        Ship Noise  Pacific Ocean  2023-01-21 15:00:00    112.036968
496        Ship Noise  Pacific Ocean  2023-01-21 16:00:00    112.658408
497  Seismic Activity   Arctic Ocean  2023-01-21 17:00:00    135.662413
498        Whale Call   Arctic Ocean  2023-01-21 18:00:00    130.228372
499  Seismic Activity   Indian Ocean  2023-01-21 19:00:00    147.700610

     depth_meters  duration_seconds  frequency_hz  water_temperature_c  \
495    451.208859             251.0   7079.977516             2.747462
496    715.730821             124.0  11681.449115            27.519407
497    439.428508             121.0   1573.138047             4.104559
498    574.209537               NaN  19488.408257            28.507121
499    529.929258             120.0           NaN            13.380173

     salinity_psu  distance_from_source_m  ambient_noise_db
495     34.677490             6603.856047         64.769764
496     34.336432             9570.484749         67.318371
497     33.244458              782.684362         75.967314
498     32.658500              664.841739         79.422180
499     36.043336             2893.652039         62.227526
```

[21]:
```
file.describe() #.describe() calculates a few summary statistics for each column
```

[21]:
```
       intensity_db  depth_meters  duration_seconds  frequency_hz  \
count    475.000000    485.000000        480.000000    450.000000
mean     120.241194    508.021042        148.018750  10284.649069
std       15.011666     99.284176         83.683204   5915.608527
min       79.546700    210.374462          1.000000    121.130445
25%      110.517344    441.831909         76.000000   5177.861299
50%      119.509579    510.537551        146.500000  10619.442047
75%      129.846824    570.835645        219.250000  15419.060989
max      166.183212    757.970934        297.000000  19859.436626

       water_temperature_c  salinity_psu  distance_from_source_m  \
count           450.000000    450.000000              450.000000
mean             14.319904     33.681080             4947.540520
```

```
std              8.608401     2.075972              2835.603933
min              0.138961    30.034580               131.860810
25%              6.768430    31.809770              2421.566842
50%             13.868108    33.844574              5054.067706
75%             21.650957    35.522021              7376.650081
max             29.991530    36.995896              9983.640363

       ambient_noise_db
count        450.000000
mean          65.118901
std            8.529912
min           50.046953
25%           58.281554
50%           65.280311
75%           72.134066
max           79.863125
```

[22]:
```
file.info() #.info() shows information on each of the columns such as the data␣
↪type and number of missing values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   sound_type              500 non-null    object
 1   location                500 non-null    object
 2   time                    500 non-null    object
 3   intensity_db            475 non-null    float64
 4   depth_meters            485 non-null    float64
 5   duration_seconds        480 non-null    float64
 6   frequency_hz            450 non-null    float64
 7   water_temperature_c     450 non-null    float64
 8   salinity_psu            450 non-null    float64
 9   distance_from_source_m  450 non-null    float64
 10  ambient_noise_db        450 non-null    float64
dtypes: float64(8), object(3)
memory usage: 43.1+ KB
```

[23]:
```
file.isnull().sum() #this will return the number of missing values in dataset
```

[23]:
```
sound_type               0
location                 0
time                     0
intensity_db            25
depth_meters            15
duration_seconds        20
```

```
frequency_hz             50
water_temperature_c      50
salinity_psu             50
distance_from_source_m   50
ambient_noise_db         50
dtype: int64
```

[24]:
```
file['depth_meters'] = file['depth_meters'].fillna(0) #this fillna() replaces␣
↪missing values in the `society` column with `'Null Values'` and then shows␣
↪the updated column.
file['depth_meters']
```

[24]:
```
0      558.392819
1      464.070791
2      559.065483
3      610.870358
4      582.048218
          …
495    451.208859
496    715.730821
497    439.428508
498    574.209537
499    529.929258
Name: depth_meters, Length: 500, dtype: float64
```

[25]:
```
file['intensity_db'] = file['intensity_db'].fillna(0) #this fillna() replaces␣
↪missing values in the `society` column with `'Null Values'` and then shows␣
↪the updated column.
file['intensity_db']
```

[25]:
```
0      125.126340
1      148.142563
2      134.256358
3      111.346445
4      106.523780
          …
495    112.036968
496    112.658408
497    135.662413
498    130.228372
499    147.700610
Name: intensity_db, Length: 500, dtype: float64
```

[26]:
```
file['duration_seconds'] = file['duration_seconds'].fillna(0) #this fillna()␣
↪replaces missing values in the `society` column with `'Null Values'` and␣
↪then shows the updated column.
file['duration_seconds']
```

[26]:
```
0      200.0
1       79.0
2      173.0
3       69.0
4       87.0
       …
495    251.0
496    124.0
497    121.0
498      0.0
499    120.0
Name: duration_seconds, Length: 500, dtype: float64
```

[27]:
```python
file['frequency_hz'] = file['frequency_hz'].fillna(0)
file['frequency_hz']
```

[27]:
```
0       7503.311575
1      19015.271842
2      14645.238957
3      11981.196514
4       3137.252436
           …
495     7079.977516
496    11681.449115
497     1573.138047
498    19488.408257
499        0.000000
Name: frequency_hz, Length: 500, dtype: float64
```

[28]:
```python
file['water_temperature_c'] = file['water_temperature_c'].fillna(0)
file['water_temperature_c']
```

[28]:
```
0      20.944851
1      16.082891
2       9.285828
3      24.413851
4      20.541935
         …
495     2.747462
496    27.519407
497     4.104559
498    28.507121
499    13.380173
Name: water_temperature_c, Length: 500, dtype: float64
```

[29]:
```python
file['salinity_psu'] = file['salinity_psu'].fillna(19)
file['salinity_psu']
```

```
[29]: 0      19.000000
      1      33.793307
      2      36.110621
      3      35.125574
      4      35.645928
               …
      495    34.677490
      496    34.336432
      497    33.244458
      498    32.658500
      499    36.043336
      Name: salinity_psu, Length: 500, dtype: float64
```

```python
[30]: file['distance_from_source_m'] = file['distance_from_source_m'].fillna(0)
      file['distance_from_source_m']
```

```
[30]: 0      5238.909673
      1      4843.900589
      2       353.856451
      3      3478.353493
      4      3863.936626
                 …
      495    6603.856047
      496    9570.484749
      497     782.684362
      498     664.841739
      499    2893.652039
      Name: distance_from_source_m, Length: 500, dtype: float64
```

```python
[31]: file['ambient_noise_db'] = file['ambient_noise_db'].fillna(0)
      file['ambient_noise_db']
```

```
[31]: 0       0.000000
      1       0.000000
      2       0.000000
      3      57.486386
      4      58.158492
               …
      495    64.769764
      496    67.318371
      497    75.967314
      498    79.422180
      499    62.227526
      Name: ambient_noise_db, Length: 500, dtype: float64
```

```python
[32]: #removing duplicates
      file.drop_duplicates()
```

[32]:
```
        sound_type       location                 time  intensity_db  \
0          Ship Noise   Indian Ocean  2023-01-01 00:00:00    125.126340
1    Seismic Activity  Pacific Ocean  2023-01-01 01:00:00    148.142563
2          Whale Call  Pacific Ocean  2023-01-01 02:00:00    134.256358
3          Ship Noise   Arctic Ocean  2023-01-01 03:00:00    111.346445
4          Ship Noise   Indian Ocean  2023-01-01 04:00:00    106.523780
..                ...            ...                  ...           ...
495        Ship Noise  Pacific Ocean  2023-01-21 15:00:00    112.036968
496        Ship Noise  Pacific Ocean  2023-01-21 16:00:00    112.658408
497  Seismic Activity   Arctic Ocean  2023-01-21 17:00:00    135.662413
498        Whale Call   Arctic Ocean  2023-01-21 18:00:00    130.228372
499  Seismic Activity   Indian Ocean  2023-01-21 19:00:00    147.700610

     depth_meters  duration_seconds  frequency_hz  water_temperature_c  \
0      558.392819             200.0   7503.311575            20.944851
1      464.070791              79.0  19015.271842            16.082891
2      559.065483             173.0  14645.238957             9.285828
3      610.870358              69.0  11981.196514            24.413851
4      582.048218              87.0   3137.252436            20.541935
..            ...               ...           ...                  ...
495    451.208859             251.0   7079.977516             2.747462
496    715.730821             124.0  11681.449115            27.519407
497    439.428508             121.0   1573.138047             4.104559
498    574.209537               0.0  19488.408257            28.507121
499    529.929258             120.0      0.000000            13.380173

     salinity_psu  distance_from_source_m  ambient_noise_db
0       19.000000             5238.909673          0.000000
1       33.793307             4843.900589          0.000000
2       36.110621              353.856451          0.000000
3       35.125574             3478.353493         57.486386
4       35.645928             3863.936626         58.158492
..            ...                     ...               ...
495     34.677490             6603.856047         64.769764
496     34.336432             9570.484749         67.318371
497     33.244458              782.684362         75.967314
498     32.658500              664.841739         79.422180
499     36.043336             2893.652039         62.227526

[500 rows x 11 columns]
```

[33]:
```python
# Check the data types of all columns
print(file.dtypes)
```

```
sound_type                 object
location                   object
time                       object
```

```
intensity_db             float64
depth_meters             float64
duration_seconds         float64
frequency_hz             float64
water_temperature_c      float64
salinity_psu             float64
distance_from_source_m   float64
ambient_noise_db         float64
dtype: object
```

[34]:

```python
#transforming catergorical data to numerical/binary
from sklearn.preprocessing import StandardScaler, LabelEncoder
data =  file['depth_meters']
label_encoder = LabelEncoder()
encoded_data = label_encoder.fit_transform(data)
print(encoded_data)
```

```
[341 151 342 415 376 325 408 419 437 357 185 253 426  86 297 146 227 432
 262 232  41 369 355 477 159 271 278 452 197 282 346 261 141 264 409  67
 250  97 424  28 126 299 450 203 128 469  32   7 315 133  69 364 277 125
  47 383 358 194 468   0  27  99 210 276 171 295  48 443   0 416 292 319
 338 317 354 434 265 365 198 442 104   0 211  34 249 103 379 110 142  13
 140   6  22 372 374 311  78 208 222  57 448 384   0 226 269   9 169 102
  71 165 466 353 124 339 441 391 236 113 361 304 387 351 406 132 433 266
 471 100 464 267 111 136 156 310 327   0 217 474 463 313 229 245 349  68
 168  18 306 356 137 451  51  31 273 405 461 139 410 212 184 385 359  23
 445   0 115 305 323 279 130 106 215 420 333 149 373   2 417  16 150   0
  45 418 138 294 209 321 239  46 399 134  25 145 447 380 152   0 155 472
 300 312 403 275 167 177 201 213 366 234 367 200 240  10 388 293 400   1
 473 191 414  66 348  65 116 470 178 270 382 324 255 296 482 206 268 407
 413 421 352  59 456  58 285  91 204 290 288 309 454 318 170 396 422  50
 344 362 162 436 188 248 182 225  62  33 453  84  72   0 114  43 458 401
 101 479 397 154   5 480  37  19 402 483 438   0 343 381 370 283 242 205
  92 166  17 196  74  60 259 440 389  24  73 393  75 173 334  77 243  42
 121 287  21 316 218 335 272 435 247 144 246 332 233 230  96 109  36 465
  49  98  94 386 163 431 105 281  85 475  55 286 350 307 180 192 231 189
 395 478 127  39 199 485  87 457 460 129 337 455 148 176 122  70 112  52
 228  89 274  26 291 377  11 298 429  54 459 308  95 207   0 238 330  80
 257  35 193  81  93 430 412 347  63 157 427 252 481 303 263 158 251 187
 363   0  88   0  14 326  61   8 301 484 221 378 241 195 390 164 280 289
 108 398 181  90 329  82   0 220 411 320 216 375 439 336 224  44  64 160
 118 179 235 328 202 322 237  12  79 190  53 345 449 428 174 446 254 153
 117 161 147 258 256 223 314 423 394  29   4 392  40 172  56  15 331 371
 123   3 131 302  30 260 219 340 244  76 425 186 214   0 143  83 183 462
  38  20 444 175 107 404 120 467 360 135 476 119 368 284]
```

[47]:
```
from sklearn.preprocessing import StandardScaler, LabelEncoder
data1 =  file['intensity_db']
label_encoder = LabelEncoder()
encoded_data1 = label_encoder.fit_transform(data1)
print(encoded_data1)
```

```
[308 460 388 128  85 328  44 457 415 161  15 424 218 418  25 126 242 250
 164 350  63 215 267 333 365  56  27 422 303 106 440   0 414 256 466 452
 197 390 355 427  77 361 403  14  53   9 192 367 435 259 443  38  17 228
 315 238   8   0  45   0 310  80 153  64 225 389  72 332 146  98 220  66
 137   0 464 249 113 284 219 203 346 373 145 131 189   4  29 426 444 196
 341 300 475 409 216  78  24 281 104  33 117  60 448 382 241 433 260  89
 437 336  65 207  87  37 385 463  36 339 116 158 127  88 253  95 293 230
 201  84 130   0 331  75 264 371   0 337 114 340 103   0  23 252 291   0
 354  20 224  52 115 251  90   0 394 129 379  54 335 430   3  97 342 206
 311 125 262 213 413 289 306 174 157 169 316 171 298 467 381 185 417 175
  10  68  11 182 244 447 302 204 376   5   0 374  32 411 307 173 352 468
 274 288 163  93 377  92 257 160 325 304 398 155 191  73 166 312 372  82
 380 425 319 461   0  51  13 434 356 227 295   0 470 269   0 369 326   0
  99 323 462 423 442 154  71 217 254 407  19   0 212 170  67  22 375 258
   0  46 184 446 194  30 199 190   1 229 202 362 459 410 193  59 473 255
 243 239 279 214 133 139 237 141 110 265 195 436   2 406 419   7 183 178
  35 101  58 450 386 421 368  55 147 327   0 366 200 177 364 321 181   0
  61 347 343 186 301  50 384 209 149 400 111  34   0 344  47 451   6 449
 283 221 140 317 233 408 266 270 179 226 299  16  41 370 272 210 245 309
 142 100 277  74 318  18 396 324 290 392 445 395  12  48 121 246 334 109
 276 105 124   0  81  40  76 402  79 474 330 275  91 363 132 268 472   0
 412 112 234 453 120 456   0 135 351 391 349  26 108 198 222 348 273  43
 313 345 338 405 378 322 223  21 320 282 294  49  62 401 232 359 247 248
 387 152 263 162 168 187 285 159 420  86 208 167 431 278 397  31 292 383
 261 404 151 428 469 180 165 432 441 148 172 188  42  83  69 102 235 287
 439  70 393 205 231 358  57 314 271 329 297 471 118 143 122 136 119 416
 429 134  94   0 138 353 280  28 438 455 123 176 296 305 357 465 211  96
  39 107 236 454 150 286 240   0   0 144 156 399 360 458]
```

[35]:
```
# Mean
mean = file['depth_meters'].mean()
print(f"Mean is  {mean}")
```

```
Mean is  492.78041088251854
```

[36]:
```
# Mode
mode = file['depth_meters'].mode()
print(f"mode is {mode}")
```

```
mode is 0    0.0
Name: depth_meters, dtype: float64
```

[37]:
```
# Standard Deviation
std_dev = file['depth_meters'].std()
print(f"standard deviation is {std_dev}")
```

```
standard deviation is 130.71484148254444
```

[38]:
```
# Median
median = file['depth_meters'].median()
print(f"median is {median}")
```

```
median is 505.8140147408641
```

[40]:
```
# Variance
variance = file['depth_meters'].var()
print(f"variance is {variance}")
```

```
variance is 17086.36978380672
```

[41]:
```
# Interquartile Range (IQR)
Q1 = file['depth_meters'].quantile(0.25)  # First Quartile
Q3 = file['depth_meters'].quantile(0.75)  # Third Quartile
IQR = Q3 - Q1  # Interquartile Range
print(IQR)
```

```
133.47197294442833
```

[42]:
```
# Maximum and Minimum
maxum = file['depth_meters'].max()
minum = file['depth_meters'].min()
range1 = maxum - minum
print(range1)
```

```
757.9709337654319
```

[43]:
```
# Coefficient of Variation (CV)
cv = std_dev / mean
print(cv)
```

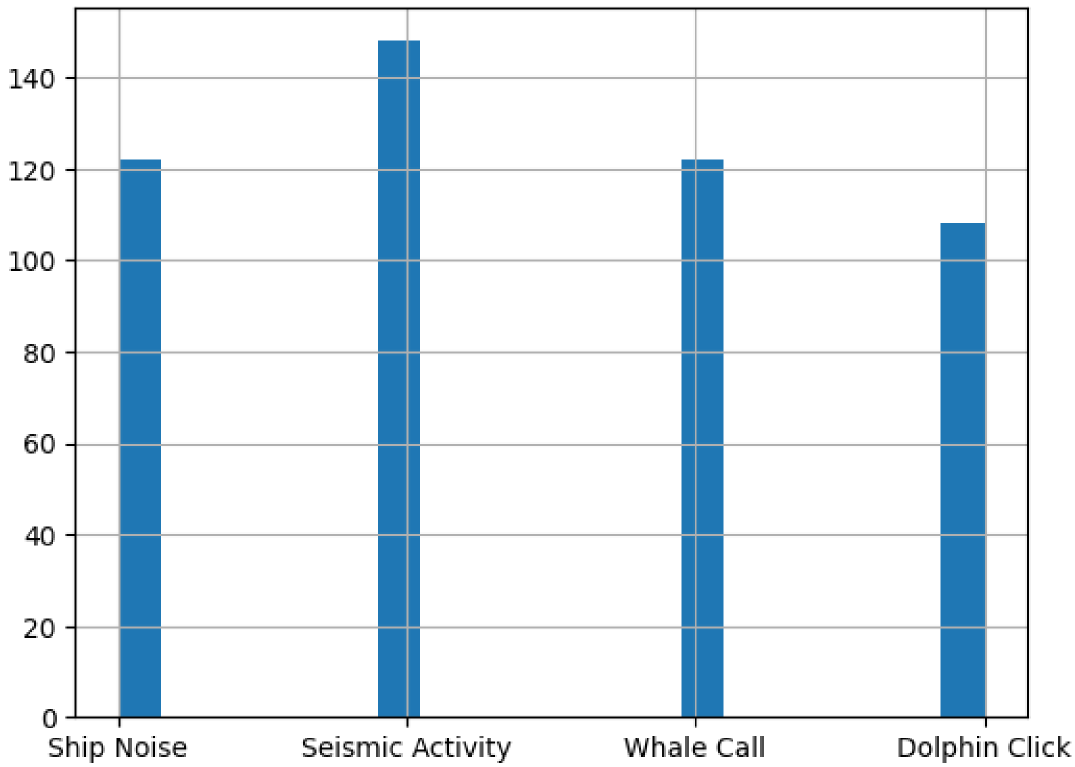
```
0.2652598167375358
```

[45]:
```
#checking if there are still any missing values
missing_values = file[['depth_meters', 'intensity_db']].isnull().sum()
print(missing_values)
```

```
depth_meters    0
intensity_db    0
dtype: int64
```

[ ]:
```
!pip install seaborn
```

[49]:
```
#understanding types of sounds
file["sound_type"].hist(bins=20)
plt.show()
```



Histogram of Sound Types:

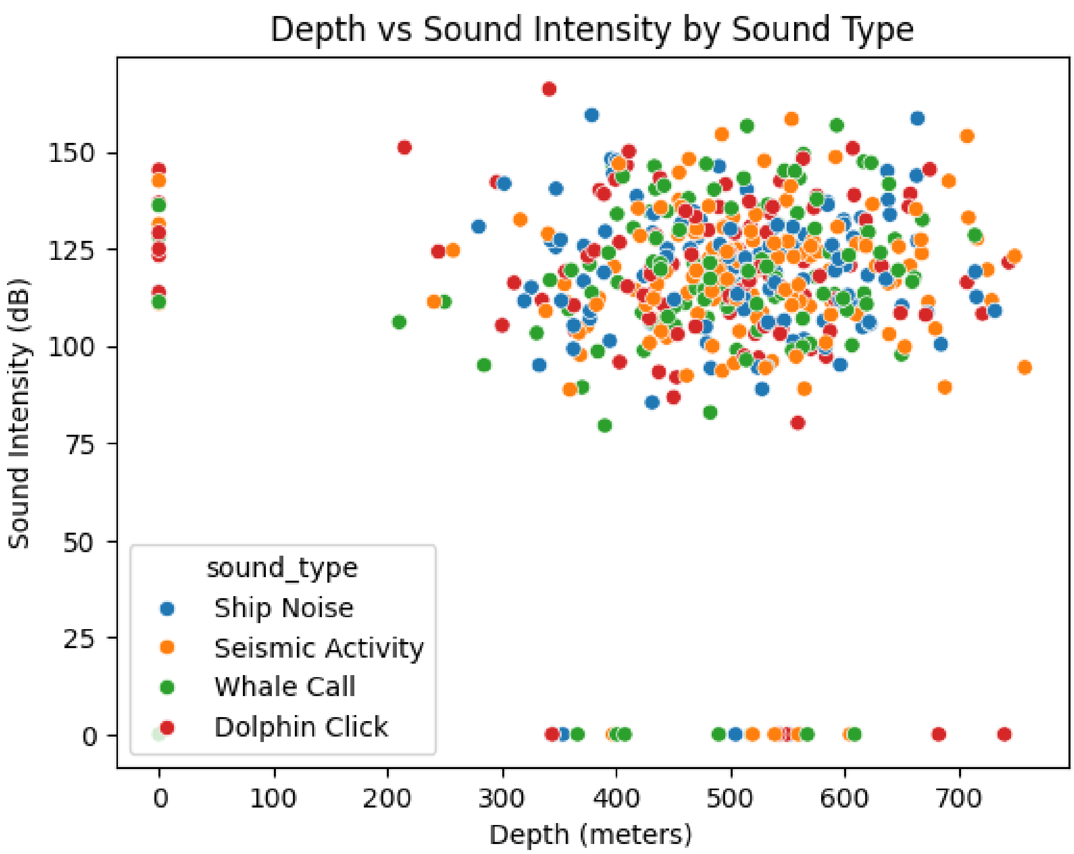
X-axis: Different types of sound available in our dataset

Y-axis: Frequency (count) of each sound type.

This histogram shows the distribution and frequency of sound types in the dataset.

[53]:
```
#Scatter plot of all types of sound available in our dataset represented by␣
↪different colours
import seaborn as sns

sns.scatterplot(x='depth_meters', y='intensity_db', hue='sound_type', data=file)
plt.xlabel('Depth (meters)')
plt.ylabel('Sound Intensity (dB)')
plt.title('Depth vs Sound Intensity by Sound Type')
plt.show()
```



Scatter Plot Explanation:

X-axis (Depth): Depth of water in meters; moving right indicates deeper water.

Y-axis (Sound Intensity): Sound intensity in decibels (dB); moving up means louder sounds.

Colored Dots: Each dot shows sound intensity at a specific depth for a sound type, with colors indicating different sounds (e.g., whale calls, ship noise).

Observations:

The graph shows sound intensity at specific depths for each sound type.

You can see how intensity changes with depth for different sounds.

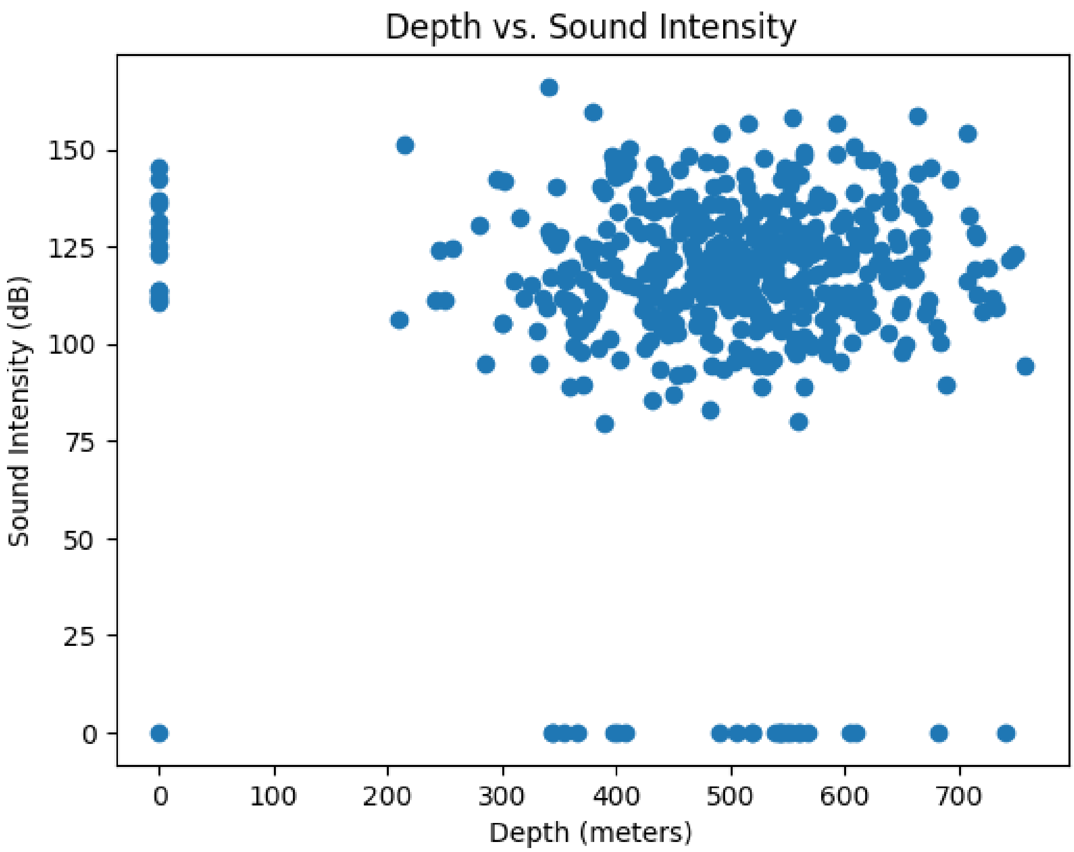
Certain sounds may be louder at specific depths.

Dots of different colors close together indicate multiple sounds at that depth.

This scatter plot illustrates the relationship between depth and sound intensity for various sound types in the underwater environment.

[48]:
```
#scatter plot to visualize how sound intensity changes with depth
import matplotlib.pyplot as plt
import seaborn as sns
```

```
plt.scatter(file['depth_meters'], file['intensity_db'])
plt.xlabel('Depth (meters)')
plt.ylabel('Sound Intensity (dB)')
plt.title('Depth vs. Sound Intensity')
plt.show()
```



Explaination of Scatter Plot of Depth vs. Sound Intensity:

X-axis: Depth of water in meters.

Y-axis: Sound intensity in decibels (dB).

Each dot represents the sound intensity at a specific depth.

This scatter plot visualizes the relationship between water depth and sound intensity.

[55]:
```
correlation = file[['depth_meters', 'intensity_db']].corr()
print(correlation)
#The correlation coefficient between depth and sound intensity is -0.003388,␣
↪indicating a
```

```
#negligible negative correlation. This suggests that changes in depth do not␣
↪significantly affect sound intensity in this dataset.
```

```
              depth_meters  intensity_db
depth_meters      1.000000     -0.003388
intensity_db     -0.003388      1.000000
```

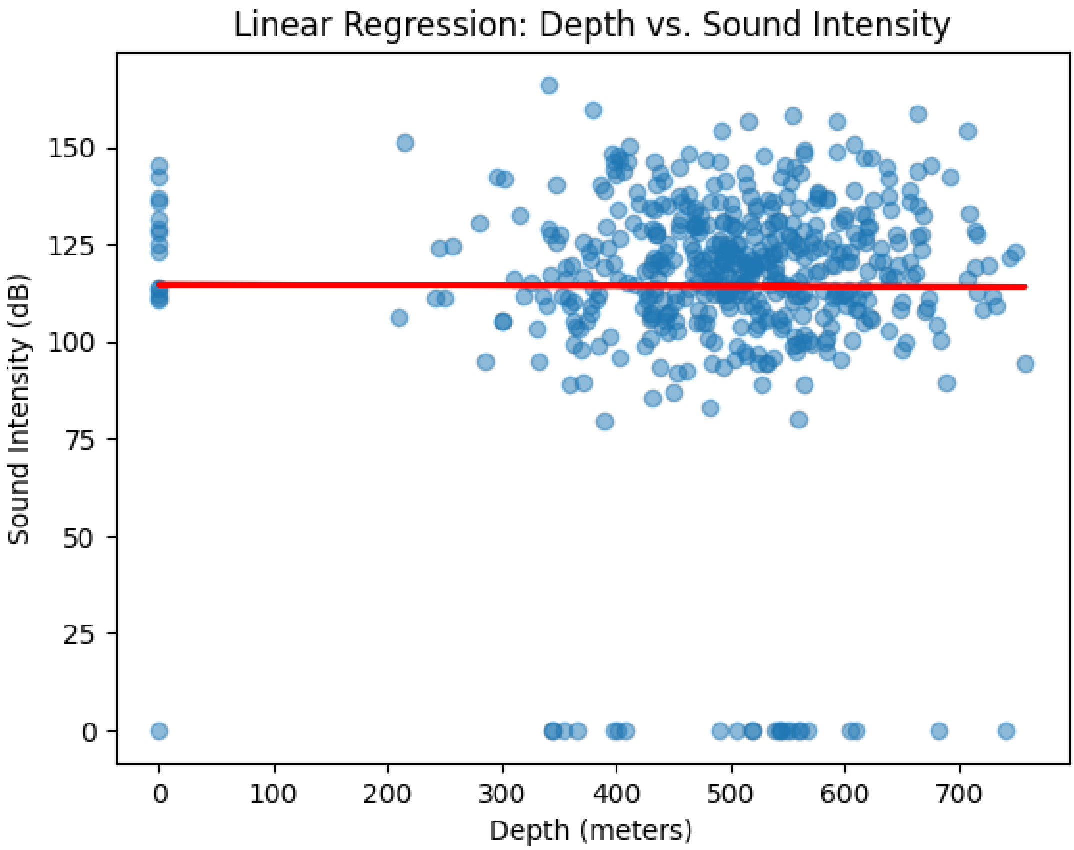
[52]:
```
#Performing linear regression to predict the relationship between depth and␣
↪sound intensity. This will help predict sound intensity based on depth.
#Import the LinearRegression class from sklearn
from sklearn.linear_model import LinearRegression

#Get access to 'depth_meters' column of our data set and 'intensity_db' column␣
↪of our dataset and store them in variable X and Y resp
X = file[['depth_meters']]  #Depth of water(input)
y = file['intensity_db']    #Sound intensity(target/output)

#Create a Linear Regression model and fit it to the data
model = LinearRegression()  #Initialize the linear regression model
model.fit(X, y)             #Train the model using the input and target data

#Predict the sound intensity based on the depth values in the data
predictions = model.predict(X)

#Plot the actual data points as a scatter plot and the regression line
plt.scatter(file['depth_meters'], file['intensity_db'], alpha=0.5)  #Plot␣
↪actual data (blue dots)
plt.plot(file['depth_meters'], predictions, color='red', linewidth=2)  #Plot␣
↪regression line (red line)
plt.xlabel('Depth (meters)')
plt.ylabel('Sound Intensity (dB)')
plt.title('Linear Regression: Depth vs. Sound Intensity')
plt.show()
```



Graph Explanation:

Blue Dots: These are the actual data points. Each blue dot shows the depth of the water (on the x-axis) and the sound intensity at that depth (on the y-axis). So, each dot represents a real measurement: "At this depth, the sound intensity was this much."

Red Line: This is the regression line. It's the best-fit straight line that shows the overall trend of how sound intensity changes with depth. It's what the linear regression model predicts.

If a blue dot is close to the red line, the model's prediction is close to the actual value.

If a blue dot is far from the red line, the model's prediction is less accurate at that depth, but the red line still captures the overall pattern.

In the graph:

The x-axis (horizontal) shows the depth underwater, and the y-axis (vertical) shows the sound intensity.

The red line helps us understand the general relationship: how sound intensity changes as depth increases.

[ ]: