INGV - Volcanic Eruption Prediction, Data Analysis.

```
In [1]: import numpy as np
        import pandas as pd
        import os
        import gc

        import matplotlib.pyplot as plt
        from plotly.offline import init_notebook_mode
        from plotly.offline import iplot, plotly
        init_notebook_mode(connected=True)
        from plotly.tools import make_subplots, tools_digest
        from plotly.graph_objs import Scatter, Scatter3d, Scatterpolar
        from plotly.graph_objs import Heatmap, Histogram2d, Contour
        from plotly.graph_objs import Histogram, Box1d, Box, Data, Layout
        from plotly.graph_objs import Figure, XAxis, YAxis, Scene
        from plotly.graph_objs import Marker, Line, ColorBar
        from plotly.graph_objs import XBins, YBins, Histogram2dContour
        from plotly.graph_objs import SmallMultiples

        %matplotlib inline
        %matplotlib inline
        %matplotlib inline
        import tqdm/tqdm_gui as tqdm
        from sklearn.preprocessing import LabelEncoder
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import mean_absolute_error
        from sklearn.feature_selection import SelectKBest
        from sklearn.feature_selection.mutual_info_regression import mutual_info_regression

        ERROR: Failed to parse "tqdm/tqdm_gui" as an extra name.
```

1. Data overview

```
In [2]: !ls -lhrt /kaggle/input/volcano-eruption-prediction/train/*.csv
        !ls -lhrt /kaggle/input/volcano-eruption-prediction/sample_submission.csv
```

Data sets Introduce in the train files:

- segment_id: the id for the seismic segment. Matching the name of the associated data file.
- time_to_eruption: The target value, the time until the next eruption.

```
In [3]: !ls -lhrt /kaggle/input/volcano-eruption-prediction/train/*.csv
        !ls -lhrt /kaggle/input/volcano-eruption-prediction/sample_submission.csv
        !ls -lhrt /kaggle/input/volcano-eruption-prediction/train/*.csv
        !ls -lhrt /kaggle/input/volcano-eruption-prediction/sample_submission.csv
        train = pd.read_csv("/kaggle/input/volcano-eruption-prediction/train.csv")
        train.shape
        train.head(10)

Out[3]: segment_id  time_to_eruption
        0  1000110             7411
        1  1000314            14444
        2  1000421            16907
        3  1000548            17197
        4  1000732            22481
        5  1000837            29843
        6  1001099            33274
        7  1001250            34314
        8  1001362            37915
        9  1001candidate      41306

        ERROR: Failed to parse "1001candidate" as an extra name.
```

```
In [4]: train.describe()

Out[4]: segment_id  time_to_eruption
        count  4.350000e+03      4.350000
        mean   2.055172e+09  25731.776092
        std    5.030291e+08  15189.666003
        min    1.000110e+09      6.954000
        25%    1.587301e+09  12666.250000
        50%    2.055962e+09  25070.000000
        75%    2.526630e+09  38723.500000
        max    3.340959e+09  55731.000000

        ERROR: Failed to parse "3.340959e+09" as an extra name.
```

Time to eruption distribution

```
In [5]: plt.hist(train.time_to_eruption,
        bins=100,
        color='red',
        alpha=0.5,
        edgecolor='black',
        linewidth=1.2)
        plt.title('Time to eruption for all volcanoes')

Out[5]:
```

Time to eruption for all volcanoes

```
In [6]: [segment_id, time_to_eruption]
        count  [4.350000e+03, 4.350000]
        mean   [2.055172e+09, 25731.7]
        std    [5.030291e+08, 15189.7]
        min    [1.000110e+09, 6.954000]
        25%    [1.587301e+09, 12666.2]
        50%    [2.055962e+09, 25070.0]
        75%    [2.526630e+09, 38723.5]
        max    [3.340959e+09, 55731.0]
        Name: time_to_eruption, dtype: float64

        ERROR: Failed to parse "[2.055962e+09, 25070.0]" as an extra name.
        ERROR: Failed to parse "[2.526630e+09, 38723.5]" as an extra name.
        ERROR: Failed to parse "[3.340959e+09, 55731.0]" as an extra name.
        ERROR: Failed to parse "Name: time_to_eruption, dtype: float64" as an extra name.
        Number of sensors: 10
        Sensor 1 is missed in 0.0% of segments
        Sensor 2 is missed in 0.0% of segments
        Sensor 3 is missed in 0.0% of segments
        Sensor 4 is missed in 0.0% of segments
        Sensor 5 is missed in 0.0% of segments
        Sensor 6 is missed in 0.0% of segments
        Sensor 7 is missed in 0.0% of segments
        Sensor 8 is missed in 0.0% of segments
        Sensor 9 is missed in 0.0% of segments
        Sensor 10 is missed in 0.0% of segments
        Number of missed sensors in training dataset
        Sensor 1 is missed in 0.0% of segments
        Sensor 2 is missed in 0.0% of segments
        Sensor 3 is missed in 0.0% of segments
        Sensor 4 is missed in 0.0% of segments
        Sensor 5 is missed in 0.0% of segments
        Sensor 6 is missed in 0.0% of segments
        Sensor 7 is missed in 0.0% of segments
        Sensor 8 is missed in 0.0% of segments
        Sensor 9 is missed in 0.0% of segments
        Sensor 10 is missed in 0.0% of segments
        Number of missed sensors in training dataset
        Number of missed sensor groups in training dataset
        Number of missed sensor groups in training dataset
        Number of missed sensor groups
```

```
In [1]: ind = [name_sensor+str(sensor), sensor]
        for sensor in range(1,4):
            if sensor == 1:
                print('Sensor 1 ID: ', name_sensor+str(sensor))
                print('Sensor 1')
                print('Sensor 1')
                print('Sensor 1')
            elif sensor == 2:
                print('Sensor 2 ID: ', name_sensor+str(sensor))
                print('Sensor 2')
                print('Sensor 2')
                print('Sensor 2')
            else:
                print('Sensor 3 ID: ', name_sensor+str(sensor))
                print('Sensor 3')
                print('Sensor 3')
                print('Sensor 3')

        ind_sensor = ind
        print('Data from sensors')
```

```
Sensor 1 ID:  Sensor1
Sensor 1
Sensor 1
Sensor 1
Sensor 2 ID:  Sensor2
Sensor 2
Sensor 2
Sensor 2
Sensor 3 ID:  Sensor3
Sensor 3
Sensor 3
Sensor 3
Data from sensors
```

3. Matrix

```
In [2]: # Import Numpy library
        import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [3]: # Import Numpy library
        import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [4]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [5]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [6]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [7]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [8]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [9]: import numpy as np
        # Define a 3x3 matrix
        A = np.array([[1, 2, 3],
                      [4, 5, 6],
                      [7, 8, 9]])
        # Print the matrix
        print("Matrix A:")
        print(A)
        # Calculate the transpose of the matrix
        A_transpose = A.T
        # Print the transposed matrix
        print("Transpose of A:")
        print(A_transpose)
        # Calculate the inverse of the matrix
        A_inverse = np.linalg.inv(A)
        # Print the inverse matrix
        print("Inverse of A:")
        print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [10]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [11]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [12]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [13]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [14]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [15]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [16]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [17]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [18]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [19]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [20]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [21]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [22]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [23]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [24]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [25]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [26]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [27]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [28]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [29]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [30]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [31]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [32]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [33]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [34]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [35]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [36]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [37]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [38]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [39]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [40]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [41]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transpose of A:")
         print(A_transpose)
         # Calculate the inverse of the matrix
         A_inverse = np.linalg.inv(A)
         # Print the inverse matrix
         print("Inverse of A:")
         print(A_inverse)
```

```
Matrix A:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Transpose of A:
[[1 4 7]
 [2 5 8]
 [3 6 9]]
Inverse of A:
[[ 4.50359963 -9.00719925  4.50359963]
 [-9.00719925 18.01439851 -9.00719925]
 [ 4.50359963 -9.00719925  4.50359963]]
```

```
In [42]: import numpy as np
         # Define a 3x3 matrix
         A = np.array([[1, 2, 3],
                       [4, 5, 6],
                       [7, 8, 9]])
         # Print the matrix
         print("Matrix A:")
         print(A)
         # Calculate the transpose of the matrix
         A_transpose = A.T
         # Print the transposed matrix
         print("Transp
```