2sggrnpf8

November 24, 2024

0.1 #Infosys Springboard Project- CytoAutoCluster

0.2 Created by Aniruddh Joshi

0.3 Loading the Dataset

[ ]:
```
import pandas as pd
```

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')

# Correct file path to your file in Google Drive
file_path = '/content/drive/MyDrive/dataset/data.csv'

# Read the file using the correct variable
data = pd.read_csv(file_path)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:
```
data.head()
```

[ ]:

	Event	Time	Cell_length	DNA1	DNA2	CD45RA	CD133	\
0	1	2693.0	22	4.391057	4.617262	0.162691	-0.029585	
1	2	3736.0	35	4.340481	4.816692	0.701349	-0.038280	
2	3	7015.0	32	3.838727	4.386369	0.603568	-0.032216	
3	4	7099.0	29	4.255806	4.830048	0.433747	-0.027611	
4	5	7700.0	25	3.976909	4.506433	-0.008809	-0.030297	

	CD19	CD22	CD11b	…	CD117	CD49d	HLA-DR	CD64	\
0	-0.006696	0.066388	-0.009184	…	0.053050	0.853505	1.664480	-0.005376	
1	-0.016654	0.074409	0.808031	…	0.089660	0.197818	0.491592	0.144814	
2	0.073855	-0.042977	-0.001881	…	0.046222	2.586670	1.308337	-0.010961	
3	-0.017661	-0.044072	0.733698	…	0.066470	1.338669	0.140523	-0.013449	
4	0.080423	0.495791	1.107627	…	-0.006223	0.180924	0.197332	0.076167	

	CD41	Viability	file_number	event_number	label	individual
0	-0.001961	0.648429	3.627711	307	1.0	1
1	0.868014	0.561384	3.627711	545	1.0	1

```
2 -0.010413   0.643337     3.627711          1726    1.0           1
3 -0.026039  -0.026523     3.627711          1766    1.0           1
4 -0.040488   0.283287     3.627711          2031    1.0           1

[5 rows x 42 columns]
```

[ ]:
```
print("Basic Structure of the Data:")
display(data)
```

```
Basic Structure of the Data:

         Event       Time  Cell_length      DNA1      DNA2    CD45RA  \
0            1    2693.00           22  4.391057  4.617262  0.162691
1            2    3736.00           35  4.340481  4.816692  0.701349
2            3    7015.00           32  3.838727  4.386369  0.603568
3            4    7099.00           29  4.255806  4.830048  0.433747
4            5    7700.00           25  3.976909  4.506433 -0.008809
…          …          …          …         …         …         …
265622  265623  707951.44           41  6.826629  7.133022  1.474081
265623  265624  708145.44           45  6.787791  7.154026  0.116755
265624  265625  708398.44           41  6.889866  7.141219  0.684921
265625  265626  708585.44           39  6.865218  7.144353  0.288761
265626  265627  709122.44           41  6.887820  7.127359  0.360753

           CD133      CD19      CD22     CD11b  …     CD117     CD49d  \
0      -0.029585 -0.006696  0.066388 -0.009184  …  0.053050  0.853505
1      -0.038280 -0.016654  0.074409  0.808031  …  0.089660  0.197818
2      -0.032216  0.073855 -0.042977 -0.001881  …  0.046222  2.586670
3      -0.027611 -0.017661 -0.044072  0.733698  …  0.066470  1.338669
4      -0.030297  0.080423  0.495791  1.107627  … -0.006223  0.180924
…            …         …         …         …  …         …         …
265622 -0.019174 -0.055620 -0.007261  0.063395  … -0.011105  0.533736
265623 -0.056213 -0.008864 -0.035158 -0.041845  …  0.143869  1.269464
265624 -0.006264 -0.026111 -0.030837 -0.034641  …  0.087102 -0.055912
265625 -0.011310 -0.048786  0.073983 -0.031787  … -0.047971  0.101955
265626  0.128604 -0.006934  0.109846  3.864711  …  0.080195  0.037962

          HLA-DR      CD64      CD41  Viability  file_number  event_number  \
0       1.664480 -0.005376 -0.001961   0.648429     3.627711           307
1       0.491592  0.144814  0.868014   0.561384     3.627711           545
2       1.308337 -0.010961 -0.010413   0.643337     3.627711          1726
3       0.140523 -0.013449 -0.026039  -0.026523     3.627711          1766
4       0.197332  0.076167 -0.040488   0.283287     3.627711          2031
…            …         …         …          …            …             …
265622  0.123758 -0.042495 -0.027971   0.236957     3.669327        102686
265623  0.047215 -0.008000 -0.025811  -0.003500     3.669327        102690
265624  0.501536  0.053884 -0.042602   0.107206     3.669327        102701
265625  6.200001  0.296877  0.192786   0.620872     3.669327        102706
```

```
265626  3.675123 -0.000878 -0.052526   0.310466     3.669327        102720

        label  individual
0         1.0           1
1         1.0           1
2         1.0           1
3         1.0           1
4         1.0           1
…         …           …
265622    NaN           2
265623    NaN           2
265624    NaN           2
265625    NaN           2
265626    NaN           2

[265627 rows x 42 columns]
```

[ ]:
```
print("\nData Information:")
display(data.info())
```

```
Data Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 265627 entries, 0 to 265626
Data columns (total 42 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   Event         265627 non-null  int64
 1   Time          265627 non-null  float64
 2   Cell_length   265627 non-null  int64
 3   DNA1          265627 non-null  float64
 4   DNA2          265627 non-null  float64
 5   CD45RA        265627 non-null  float64
 6   CD133         265627 non-null  float64
 7   CD19          265627 non-null  float64
 8   CD22          265627 non-null  float64
 9   CD11b         265627 non-null  float64
 10  CD4           265627 non-null  float64
 11  CD8           265627 non-null  float64
 12  CD34          265627 non-null  float64
 13  Flt3          265627 non-null  float64
 14  CD20          265627 non-null  float64
 15  CXCR4         265627 non-null  float64
 16  CD235ab       265627 non-null  float64
 17  CD45          265627 non-null  float64
 18  CD123         265627 non-null  float64
 19  CD321         265627 non-null  float64
 20  CD14          265627 non-null  float64
```

```
 21  CD33           265627 non-null  float64
 22  CD47           265627 non-null  float64
 23  CD11c          265627 non-null  float64
 24  CD7            265627 non-null  float64
 25  CD15           265627 non-null  float64
 26  CD16           265627 non-null  float64
 27  CD44           265627 non-null  float64
 28  CD38           265627 non-null  float64
 29  CD13           265627 non-null  float64
 30  CD3            265627 non-null  float64
 31  CD61           265627 non-null  float64
 32  CD117          265627 non-null  float64
 33  CD49d          265627 non-null  float64
 34  HLA-DR         265627 non-null  float64
 35  CD64           265627 non-null  float64
 36  CD41           265627 non-null  float64
 37  Viability      265627 non-null  float64
 38  file_number    265627 non-null  float64
 39  event_number   265627 non-null  int64
 40  label          104184 non-null  float64
 41  individual     265627 non-null  int64
dtypes: float64(38), int64(4)
memory usage: 85.1 MB
```

None

[ ]:
```
df=data
print("\nMissing Values:")
missing_values = df.isnull().sum()
missing_percentage = (missing_values / len(df)) * 100
missing_df = pd.DataFrame({'Missing Values': missing_values, 'Percentage':␣
 ↪missing_percentage})
display(missing_df[missing_df['Missing Values'] > 0])
```

Missing Values:

	Missing Values	Percentage
label	161443	60.778084

[ ]:
```
print("\nDescriptive Statistics:")
display(df.describe())
```

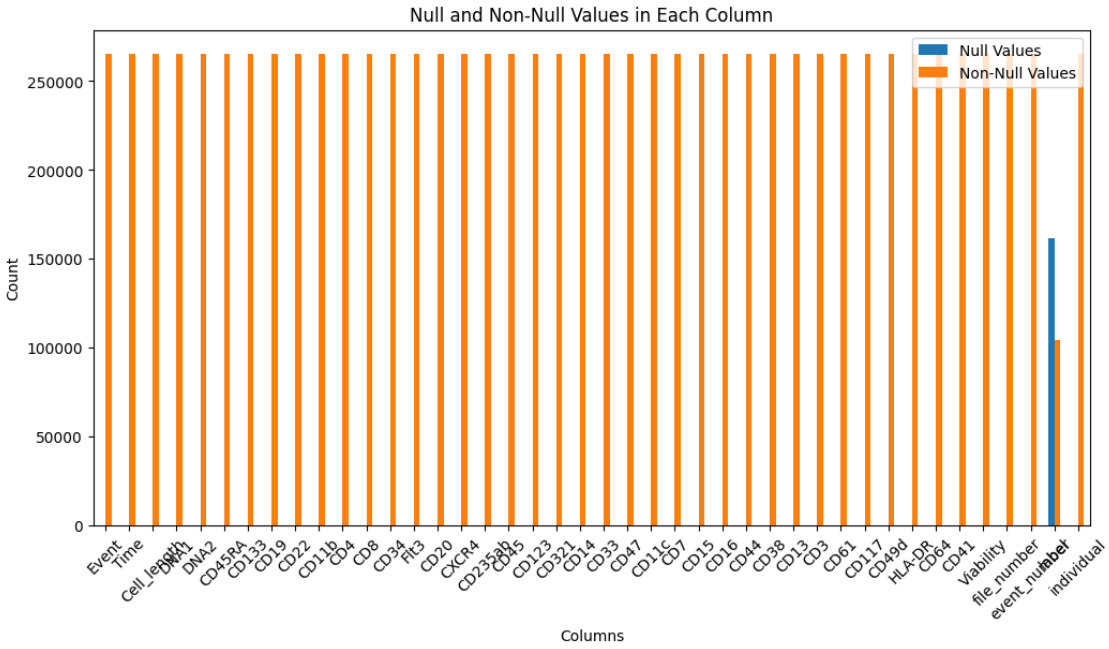
Descriptive Statistics:

	Event	Time	Cell_length	DNA1	\
count	265627.000000	265627.000000	265627.000000	265627.000000	
mean	132814.000000	272948.345014	34.450572	4.606956	

```
std     76680.054314  171220.139430      11.446694       1.312831
min         1.000000       1.000000      10.000000       2.786488
25%     66407.500000  120196.000000      26.000000       3.700023
50%    132814.000000  253276.000000      33.000000       4.022127
75%    199220.500000  424502.500000      41.000000       6.353313
max    265627.000000  709122.440000      65.000000       7.001489

                DNA2         CD45RA          CD133           CD19  \
count  265627.000000  265627.000000  265627.000000  265627.000000
mean        5.198308       0.688127       0.145960       0.509301
std         1.150357       0.609105       0.259267       0.857462
min         2.236450      -0.057305      -0.058081      -0.058089
25%         4.407822       0.204625      -0.022935      -0.018838
50%         4.698415       0.549387       0.025353       0.075210
75%         6.766268       1.031198       0.224299       0.548386
max         7.472308       6.691197       5.527494       4.990085

                CD22          CD11b  …          CD117          CD49d  \
count  265627.000000  265627.000000  …  265627.000000  265627.000000
mean        0.397323       0.710319  …       0.131199       0.794938
std         0.762126       1.011434  …       0.313208       0.627619
min        -0.057342      -0.058236  …      -0.057668      -0.058064
25%        -0.020689      -0.000294  …      -0.023957       0.283013
50%         0.058790       0.257923  …      -0.000410       0.677212
75%         0.386481       0.923517  …       0.154736       1.190787
max         5.160477       5.260789  …       5.502125       5.153438

              HLA-DR           CD64           CD41      Viability  \
count  265627.000000  265627.000000  265627.000000  265627.000000
mean        1.521812       0.551512       0.261754       0.570037
std         1.694211       0.888739       0.617065       0.589738
min        -0.057974      -0.058199      -0.058244      -0.057979
25%         0.057709      -0.010582      -0.020166       0.065523
50%         0.611335       0.122493       0.052229       0.398230
75%         2.888240       0.604131       0.305591       0.931058
max         7.052507       4.517843       7.718288       2.433031

         file_number   event_number          label     individual
count  265627.000000  265627.000000  104184.000000  265627.000000
mean        3.639348  171288.314234       8.116102       1.279625
std         0.018678  123904.361456       2.457486       0.448816
min         3.627711       1.000000       1.000000       1.000000
25%         3.627711   58679.500000       7.000000       1.000000
50%         3.627711  152783.000000       8.000000       1.000000
75%         3.669327  282369.000000      10.000000       2.000000
max         3.669327  400112.000000      14.000000       2.000000

[8 rows x 42 columns]
```

##NULL VS NOT NULL

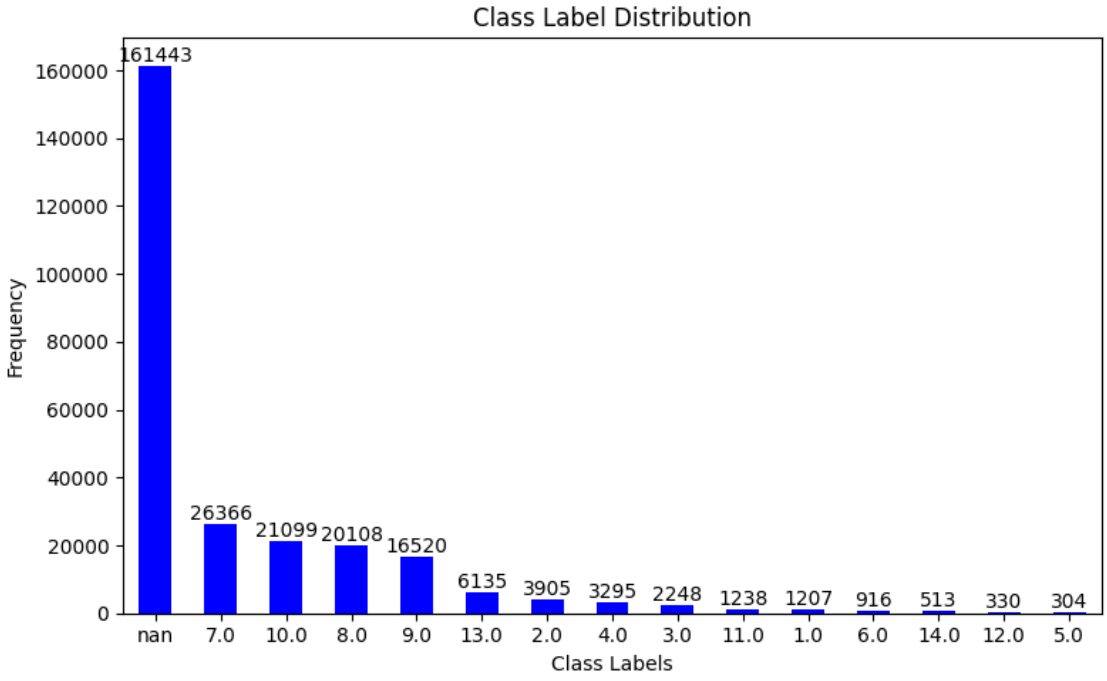
[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt

#data

null_counts = df.isnull().sum()
non_null_counts = df.notnull().sum()


plot_data = pd.DataFrame({
    'Null Values': null_counts,
    'Non-Null Values': non_null_counts
})

plot_data.plot(kind='bar', figsize=(12, 6))
plt.title('Null and Non-Null Values in Each Column')
plt.xlabel('Columns')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(loc='upper right')
plt.show()
```



##CLASS LABEL DISTRIBUTION

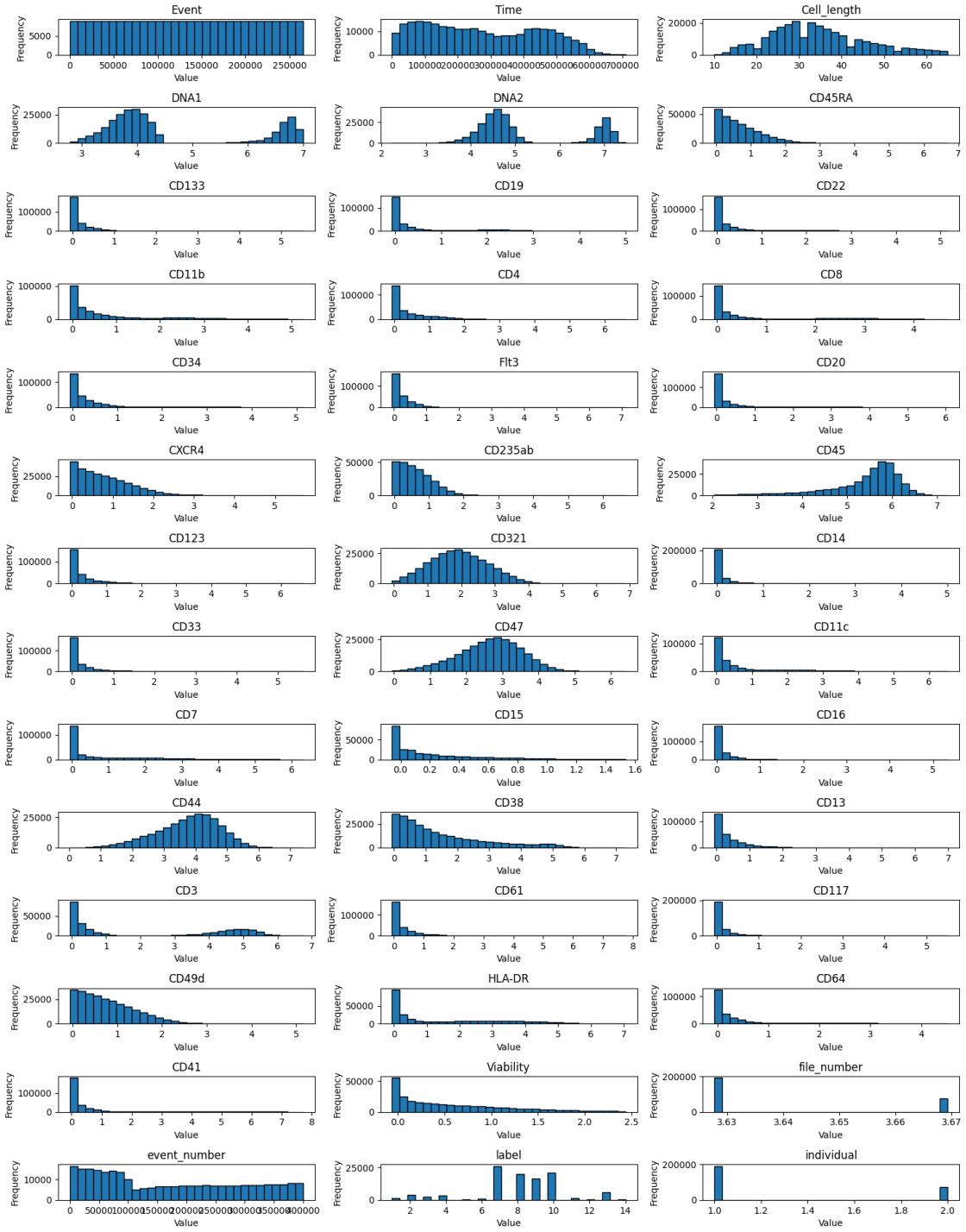
[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt

data = df


label_distribution = df['label'].value_counts(dropna=False)
print("Class Label Distribution:")
print(label_distribution)


label_distribution = df['label'].value_counts(dropna=False)

plt.figure(figsize=(8, 5))
bars = label_distribution.plot(kind='bar', color='blue')
plt.title('Class Label Distribution')
plt.xlabel('Class Labels')
plt.ylabel('Frequency')
plt.xticks(rotation=0)

for bar in bars.patches:
    bars.annotate(bar.get_height(),
                  (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                  ha='center',
                  va='bottom')

plt.tight_layout()
plt.show()
```

```
Class Label Distribution:
label
NaN     161443
7.0      26366
10.0     21099
8.0      20108
9.0      16520
13.0      6135
2.0       3905
4.0       3295
3.0       2248
11.0      1238
1.0       1207
6.0        916
14.0       513
12.0       330
5.0        304
Name: count, dtype: int64
```



##Histograms of Features

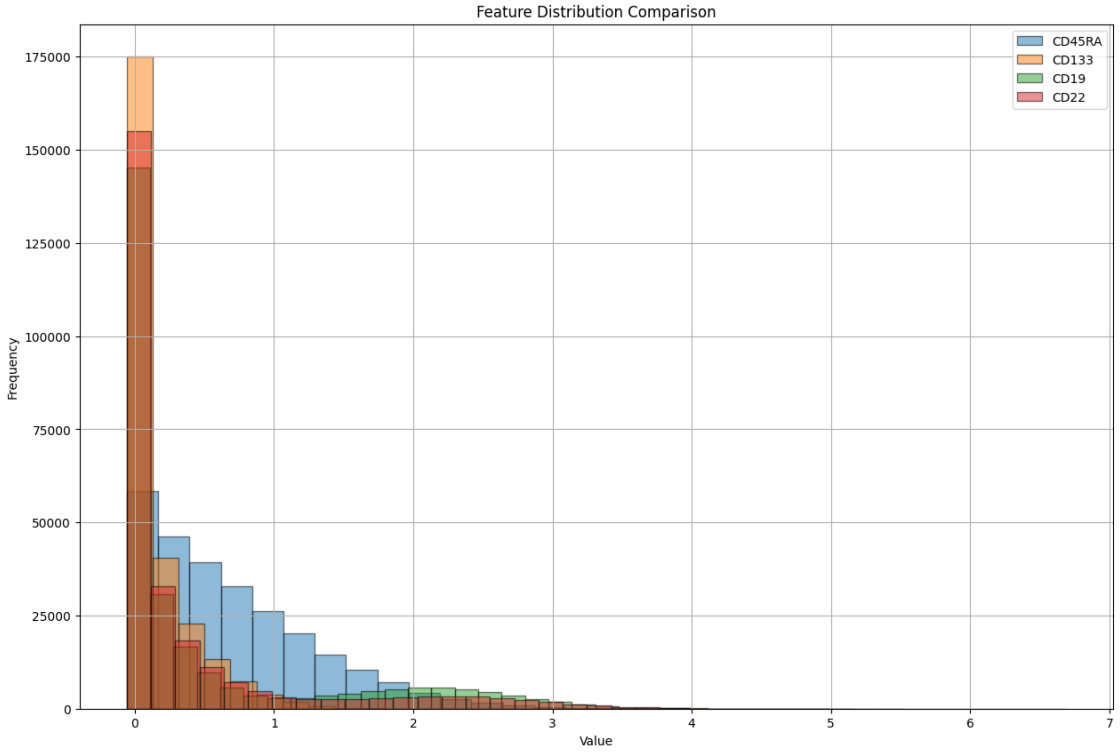
[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt

# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Select only numerical columns for histogram plotting
numerical_columns = data.select_dtypes(include=['float64', 'int64']).columns

# Set up the figure for subplots
plt.figure(figsize=(15, 20))

# Iterate through numerical columns and create a histogram for each
for i, column in enumerate(numerical_columns, 1):
    plt.subplot(len(numerical_columns)//3 + 1, 3, i)
    plt.hist(data[column], bins=30, edgecolor='black')
    plt.title(column)
    plt.xlabel('Value')
    plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

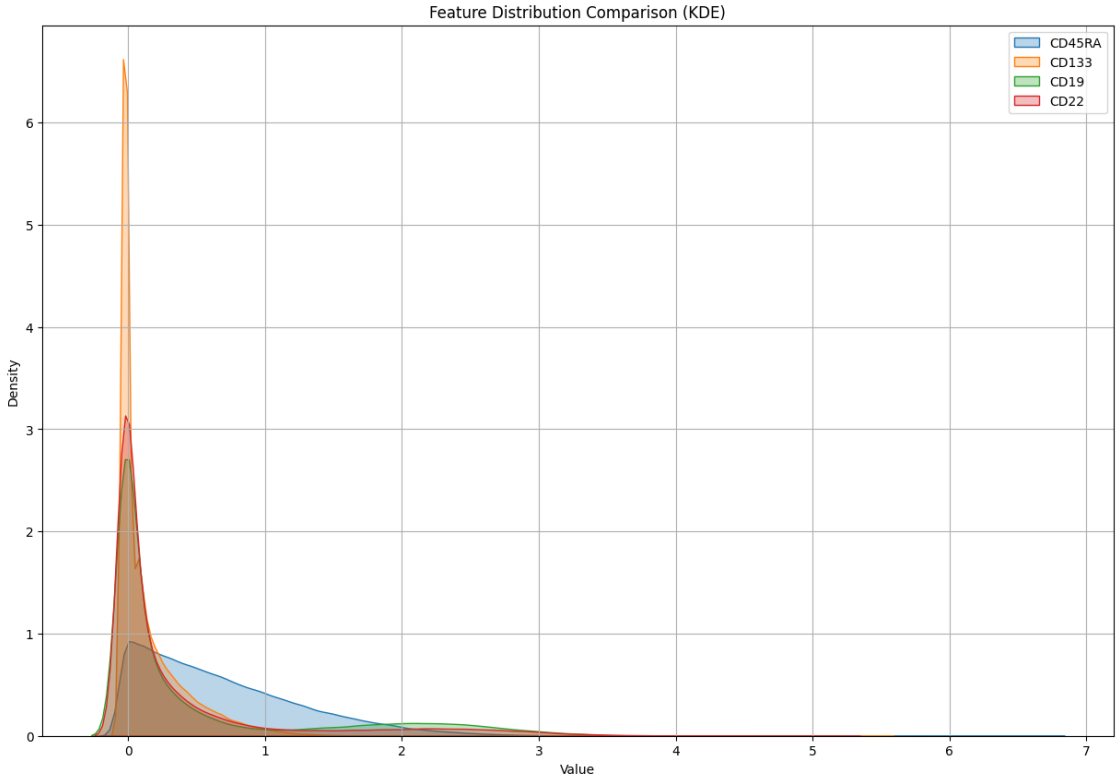


##Comparing Feature Distributions with Histograms and KDE Plots

[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

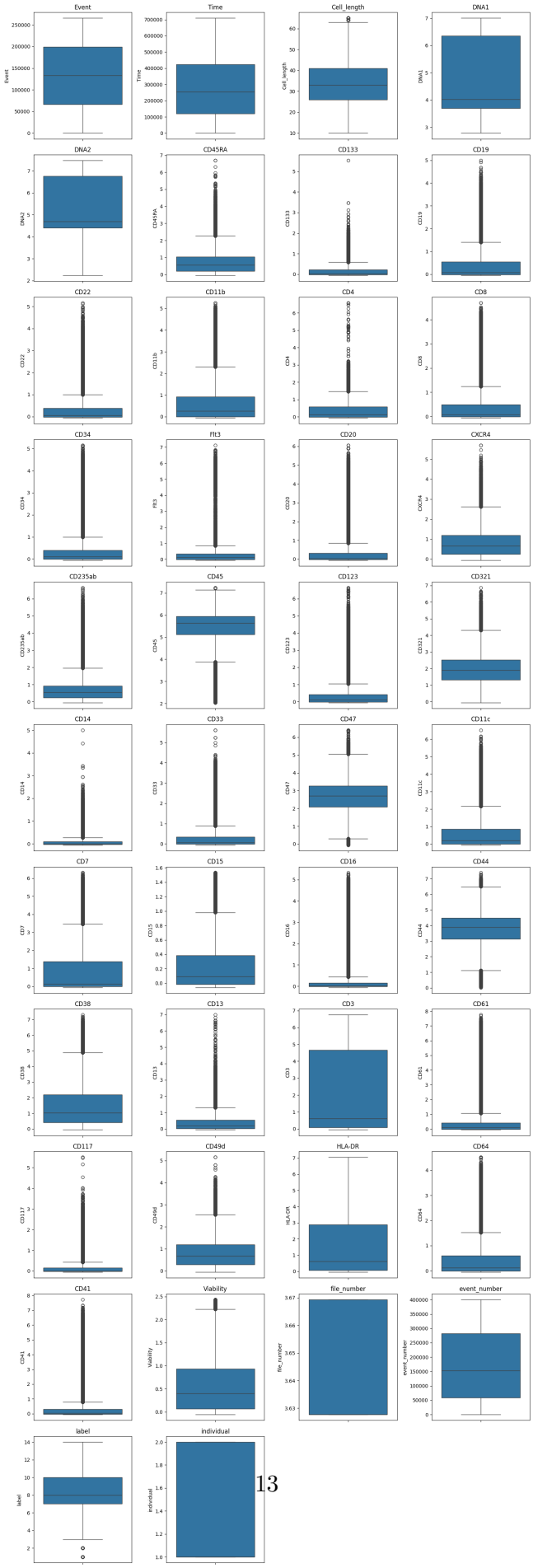
```
# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Select features for comparison (adjust based on your dataset)
features_to_compare = ['CD45RA', 'CD133', 'CD19', 'CD22']  # Example features, replace with your own
colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728']  # Custom color palette

# Step 1: Histograms for feature distribution comparison
plt.figure(figsize=(15, 10))

for feature, color in zip(features_to_compare, colors):
    plt.hist(data[feature], bins=30, alpha=0.5, label=feature, edgecolor='black', color=color)

plt.title('Feature Distribution Comparison')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.legend()
plt.grid(True)
plt.show()

# Step 2: Kernel Density Estimation (KDE) for smoother distribution comparison
plt.figure(figsize=(15, 10))

for feature, color in zip(features_to_compare, colors):
    sns.kdeplot(data[feature], label=feature, fill=True, alpha=0.3, color=color)

plt.title('Feature Distribution Comparison (KDE)')
plt.xlabel('Value')
plt.ylabel('Density')
plt.legend()
plt.grid(True)
plt.show()
```


Feature Distribution Comparison
CD45RA
CD133
CD19
CD22
Frequency
Value
0
25000
50000
75000
100000
125000
150000
175000
0
1
2
3
4
5
6
7


Feature Distribution Comparison (KDE)
CD45RA
CD133
CD19
CD22
Density
Value
0
1
2
3
4
5
6
0
1
2
3
4
5
6
7

##Box Plot Analysis of Feature Distributions

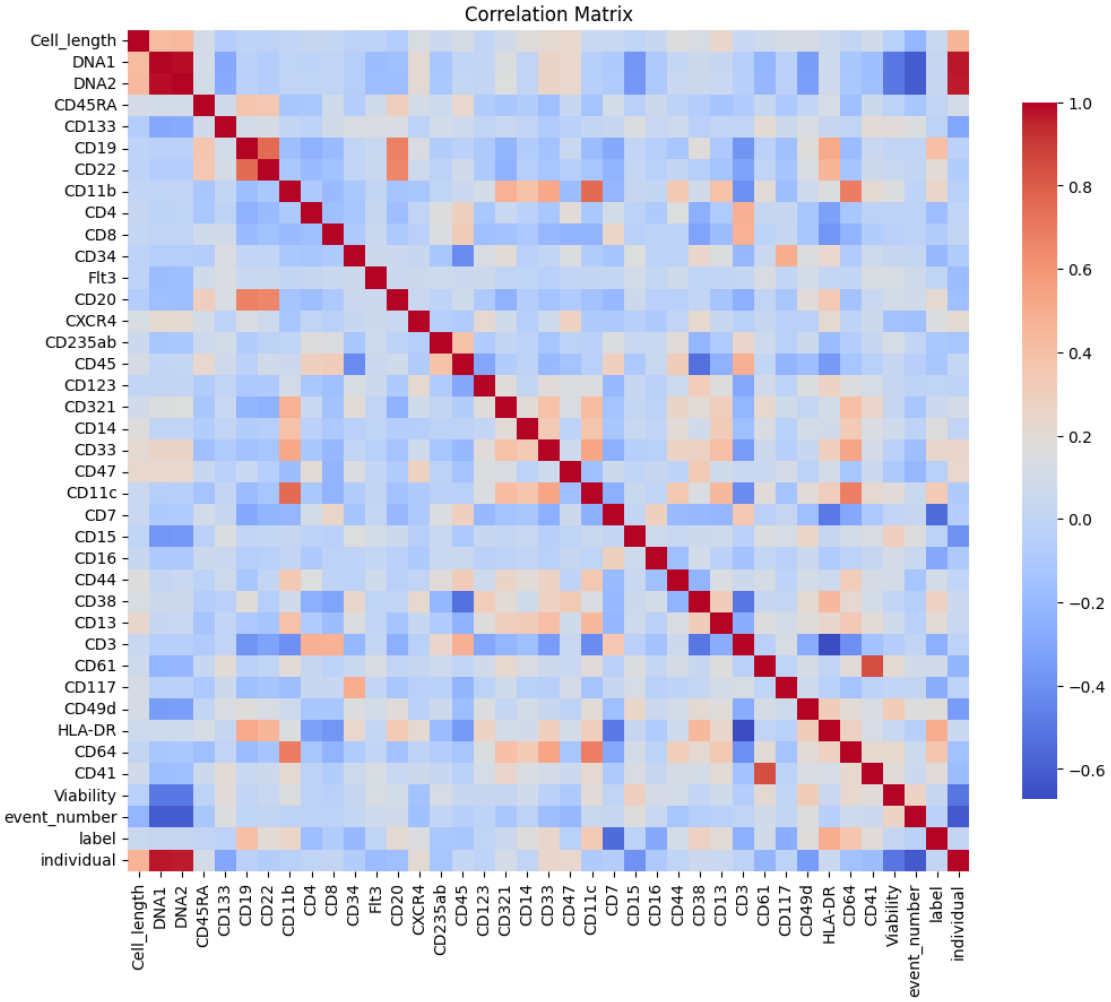
[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Step 1: Box Plots for Numerical Features
numerical_features = data.select_dtypes(include=['float64', 'int64']).columns
 ↪# Select numerical columns
rows = (len(numerical_features) // 4) + 1  # Calculate the number of rows needed

plt.figure(figsize=(15, rows * 4))
for i, feature in enumerate(numerical_features):
    plt.subplot(rows, 4, i + 1)
    sns.boxplot(data[feature])
    plt.title(feature)
plt.tight_layout()
plt.show()

# Step 2: Count Plots for Categorical Features
categorical_features = data.select_dtypes(include=['object']).columns  # Select
 ↪categorical columns

plt.figure(figsize=(15, 10))
for i, feature in enumerate(categorical_features):
    plt.subplot(2, 2, i + 1)
    sns.countplot(x=data[feature], order=data[feature].value_counts().index)
    plt.title(feature)
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


Event
Time
Cell_length
DNA1
DNA2
CD45RA
CD133
CD19
CD22
CD11b
CD4
CD8
CD34
Flt3
CD20
CXCR4
CD235ab
CD45
CD123
CD321
CD14
CD33
CD47
CD11c
CD7
CD15
CD16
CD44
CD38
CD13
CD3
CD61
CD117
CD49d
HLA-DR
CD64
CD41
Viability
file_number
event_number
label
individual

```
<Figure size 1500x1000 with 0 Axes>
```

##Feature Correlation Matrix Analysis

[ ]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Drop the specified columns
data = data.drop(columns=['file_number', 'Event', 'Time'])

# Calculate the correlation matrix
correlation_matrix = data.corr()

# Set up the matplotlib figure
plt.figure(figsize=(12, 10))

# Create a heatmap using Seaborn without annotations
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', square=True,␣
 ↪cbar_kws={"shrink": .8})

plt.title('Correlation Matrix')
plt.show()
```



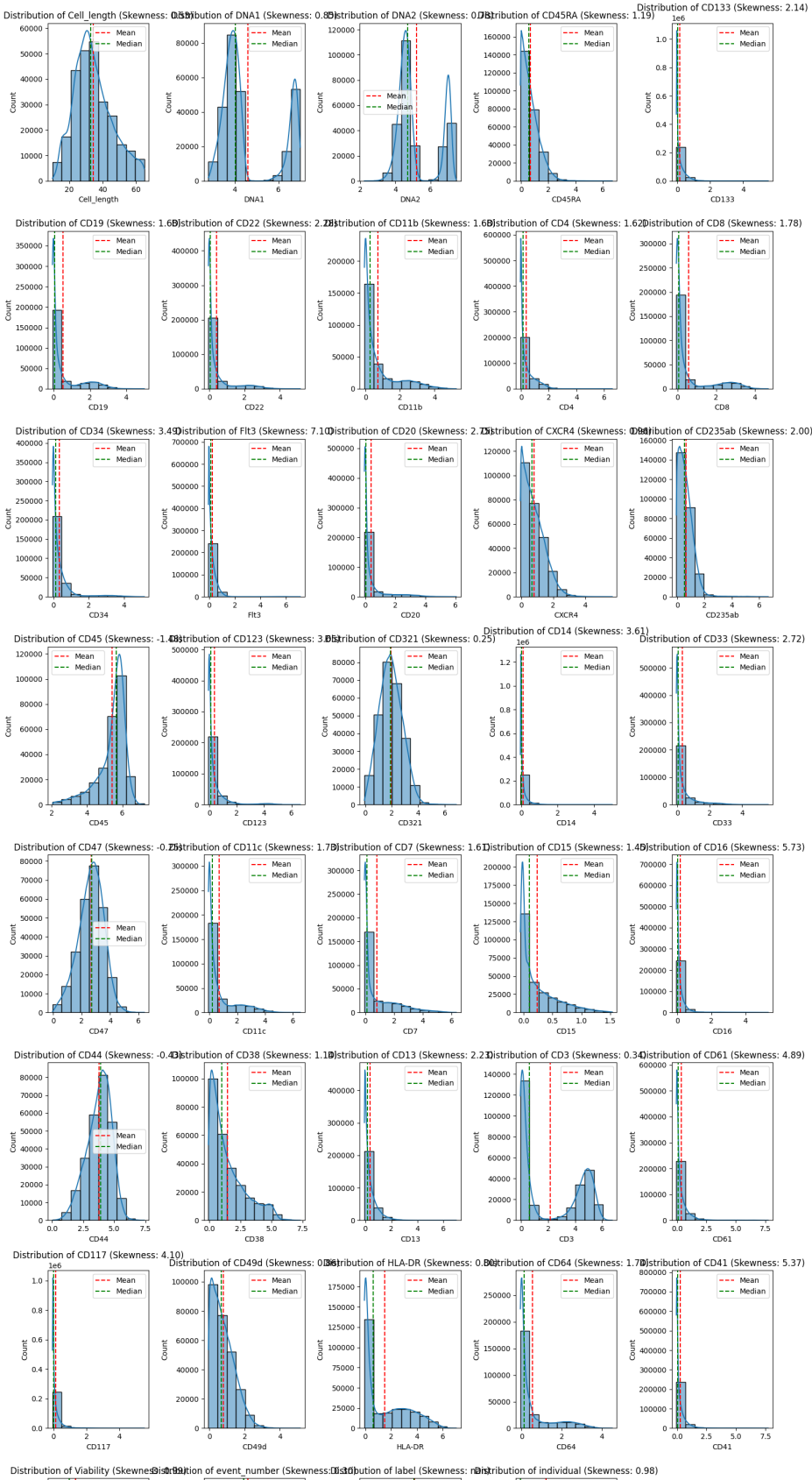
##Analysis of Feature Skewness

[ ]:
```
import pandas as pd
from scipy.stats import skew
import matplotlib.pyplot as plt
import seaborn as sns
import math

# Load the data
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')
data = data.drop(columns=['file_number', 'Event', 'Time'])

# Calculate skewness
skewness = data.apply(skew)

# Function to categorize skewness
```

```
def categorize_skewness(value):
    if value > 0.5:
        return 'Right-skewed'
    elif value < -0.5:
        return 'Left-skewed'
    else:
        return 'Approximately symmetrical'

# Apply the categorization
skewness_category = skewness.apply(categorize_skewness)

# Display skewness and its categorization
skewness_df = pd.DataFrame({'Skewness': skewness, 'Category':␣
 ↪skewness_category})
print(skewness_df)

# Set the number of columns in the grid
n_cols = 5  # Adjust this value for number of plots per row
n_plots = len(data.columns)
n_rows = math.ceil(n_plots / n_cols)

# Create subplots grid
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 4 * n_rows))  # Adjust␣
 ↪figsize for larger or smaller plots
axes = axes.flatten()  # Flatten axes array to make it easier to index

# Loop through columns and plot histograms on each subplot
for idx, col in enumerate(data.columns):
    sns.histplot(data[col], bins=10, kde=True, ax=axes[idx])
    axes[idx].set_title(f'Distribution of {col} (Skewness: {skewness[col]:.
 ↪2f})')
    axes[idx].axvline(data[col].mean(), color='red', linestyle='--',␣
 ↪label='Mean')
    axes[idx].axvline(data[col].median(), color='green', linestyle='--',␣
 ↪label='Median')
    axes[idx].legend()

# Remove any unused subplots (if n_plots is not a perfect multiple of n_cols)
for i in range(n_plots, len(axes)):
    fig.delaxes(axes[i])

# Ensure the layout is tight and the plot is shown properly
plt.tight_layout()
plt.show(block=True)  # Ensure plt.show() does not block rendering
```

	Skewness	Category
Cell_length	0.527832	Right-skewed

DNA1	0.845010	Right-skewed
DNA2	0.779167	Right-skewed
CD45RA	1.191595	Right-skewed
CD133	2.141953	Right-skewed
CD19	1.682609	Right-skewed
CD22	2.283181	Right-skewed
CD11b	1.679089	Right-skewed
CD4	1.622044	Right-skewed
CD8	1.775713	Right-skewed
CD34	3.492437	Right-skewed
Flt3	7.098151	Right-skewed
CD20	2.754699	Right-skewed
CXCR4	0.955342	Right-skewed
CD235ab	2.001479	Right-skewed
CD45	-1.484824	Left-skewed
CD123	3.648890	Right-skewed
CD321	0.247097	Approximately symmetrical
CD14	3.609006	Right-skewed
CD33	2.724977	Right-skewed
CD47	-0.250323	Approximately symmetrical
CD11c	1.733888	Right-skewed
CD7	1.606528	Right-skewed
CD15	1.445147	Right-skewed
CD16	5.733203	Right-skewed
CD44	-0.431589	Approximately symmetrical
CD38	1.141482	Right-skewed
CD13	2.234311	Right-skewed
CD3	0.342239	Approximately symmetrical
CD61	4.894707	Right-skewed
CD117	4.097508	Right-skewed
CD49d	0.856805	Right-skewed
HLA-DR	0.795359	Right-skewed
CD64	1.743733	Right-skewed
CD41	5.366314	Right-skewed
Viability	0.985417	Right-skewed
event_number	0.304116	Approximately symmetrical
label	NaN	Approximately symmetrical
individual	0.982030	Right-skewed


Distribution of Cell_length (Skewness: 0.53)
Distribution of DNA1 (Skewness: 0.85)
Distribution of DNA2 (Skewness: 0.78)
Distribution of CD45RA (Skewness: 1.19)
Distribution of CD133 (Skewness: 2.14)
Distribution of CD19 (Skewness: 1.68)
Distribution of CD22 (Skewness: 2.28)
Distribution of CD11b (Skewness: 1.68)
Distribution of CD4 (Skewness: 1.62)
Distribution of CD8 (Skewness: 1.78)
Distribution of CD34 (Skewness: 3.49)
Distribution of Flt3 (Skewness: 7.10)
Distribution of CD20 (Skewness: 2.25)
Distribution of CXCR4 (Skewness: 0.94)
Distribution of CD235ab (Skewness: 2.00)
Distribution of CD45 (Skewness: -1.48)
Distribution of CD123 (Skewness: 3.05)
Distribution of CD321 (Skewness: 0.25)
Distribution of CD14 (Skewness: 3.61)
Distribution of CD33 (Skewness: 2.72)
Distribution of CD47 (Skewness: -0.25)
Distribution of CD11c (Skewness: 1.78)
Distribution of CD7 (Skewness: 1.61)
Distribution of CD15 (Skewness: 1.48)
Distribution of CD16 (Skewness: 5.73)
Distribution of CD44 (Skewness: -0.43)
Distribution of CD38 (Skewness: 1.18)
Distribution of CD13 (Skewness: 2.21)
Distribution of CD3 (Skewness: 0.34)
Distribution of CD61 (Skewness: 4.89)
Distribution of CD117 (Skewness: 4.10)
Distribution of CD49d (Skewness: 0.96)
Distribution of HLA-DR (Skewness: 0.80)
Distribution of CD64 (Skewness: 1.70)
Distribution of CD41 (Skewness: 5.37)
Distribution of Viability (Skewness: 1.09)
Distribution of event_number (Skewness: 0.10)
Distribution of label (Skewness: nan)
Distribution of individual (Skewness: 0.98)
Mean
Median
Count
Cell_length
DNA1
DNA2
CD45RA
CD133
CD19
CD22
CD11b
CD4
CD8
CD34
Flt3
CD20
CXCR4
CD235ab
CD45
CD123
CD321
CD14
CD33
CD47
CD11c
CD7
CD15
CD16
CD44
CD38
CD13
CD3
CD61
CD117
CD49d
HLA-DR
CD64
CD41
Viability
event_number
label
individual

[ ]:

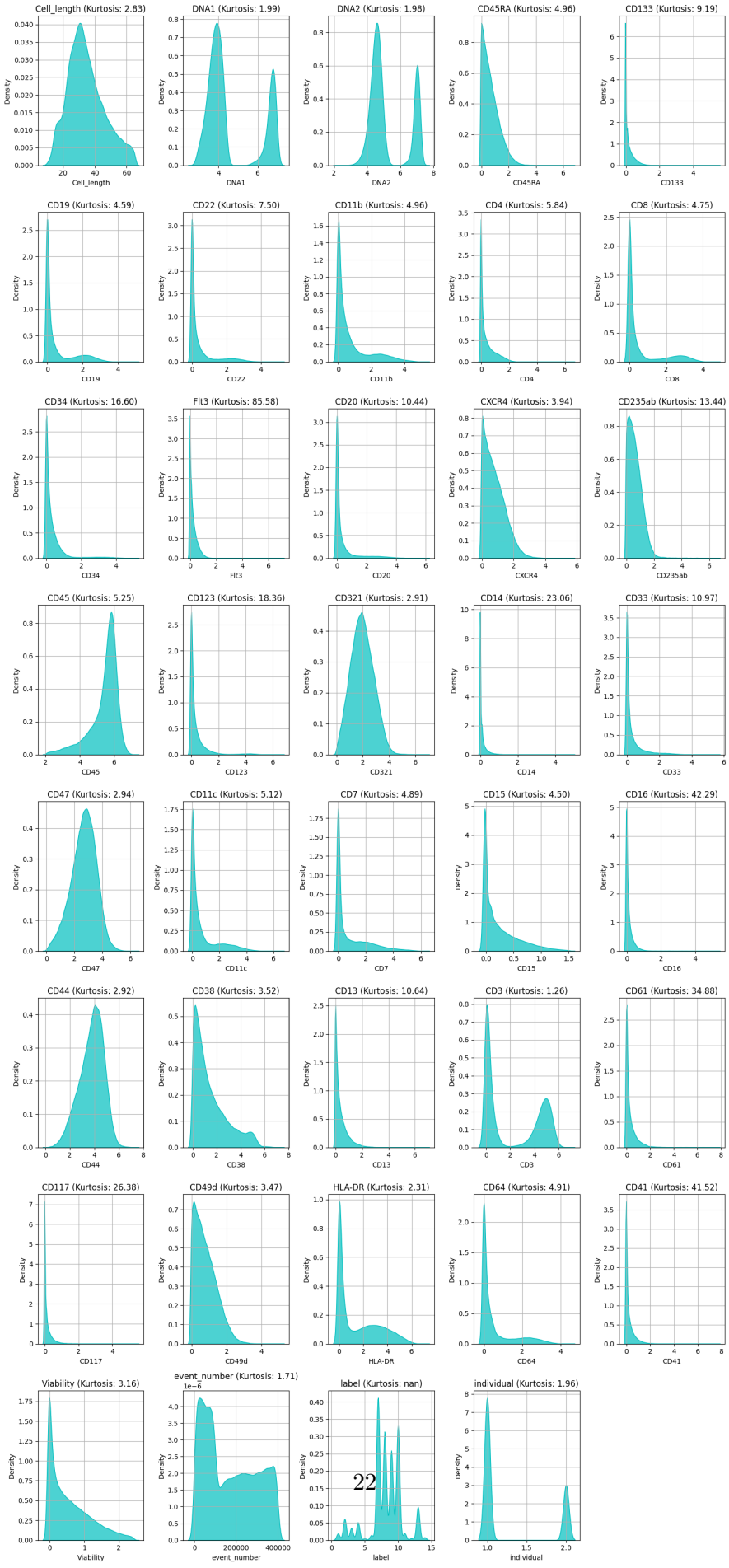
##Analysis of Feature Kurtosis

[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import kurtosis
import math

# Load the data
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Drop the specified columns
data = data.drop(columns=['file_number', 'Event', 'Time'])

# Calculate kurtosis for each column
kurtosis_values = data.apply(kurtosis, fisher=False)  # Fisher=False gives␣
 ↪Pearson kurtosis (normal kurtosis = 3)

# Create a DataFrame with kurtosis values
kurtosis_df = pd.DataFrame({'Column': data.columns, 'Kurtosis':␣
 ↪kurtosis_values})

# Categorize the kurtosis values (Leptokurtic, Mesokurtic, Platykurtic)
def categorize_kurtosis(value):
    if value > 3:
        return 'Leptokurtic (heavy tails)'
    elif value < 3:
        return 'Platykurtic (light tails)'
    else:
        return 'Mesokurtic (normal tails)'

kurtosis_df['Category'] = kurtosis_df['Kurtosis'].apply(categorize_kurtosis)

# Print the kurtosis values and their categories
print(kurtosis_df)

# Set the number of columns in the grid
n_cols = 5  # You can adjust this to control how many plots per row
n_plots = len(data.columns)
n_rows = math.ceil(n_plots / n_cols)

# Create subplots grid
```

```
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 4 * n_rows))  # Adjust␣
 ↪figsize for larger or smaller plots
axes = axes.flatten()  # Flatten axes array to make it easier to index

# Loop through columns and plot KDE on each subplot
for idx, column in enumerate(data.columns):
    sns.kdeplot(data[column].dropna(), color='c', fill=True, alpha=0.7,␣
 ↪ax=axes[idx])
    axes[idx].set_title(f'{column} (Kurtosis: {kurtosis_df.
 ↪loc[kurtosis_df["Column"] == column, "Kurtosis"].values[0]:.2f})')
    axes[idx].set_xlabel(column)
    axes[idx].set_ylabel('Density')
    axes[idx].grid(True)

# Remove any unused subplots (if n_plots is not a perfect multiple of n_cols)
for i in range(n_plots, len(axes)):
    fig.delaxes(axes[i])

plt.tight_layout()
plt.show()
```

	Column	Kurtosis	Category
Cell_length	Cell_length	2.834033	Platykurtic (light tails)
DNA1	DNA1	1.994037	Platykurtic (light tails)
DNA2	DNA2	1.975021	Platykurtic (light tails)
CD45RA	CD45RA	4.964272	Leptokurtic (heavy tails)
CD133	CD133	9.190066	Leptokurtic (heavy tails)
CD19	CD19	4.590887	Leptokurtic (heavy tails)
CD22	CD22	7.500223	Leptokurtic (heavy tails)
CD11b	CD11b	4.964495	Leptokurtic (heavy tails)
CD4	CD4	5.844261	Leptokurtic (heavy tails)
CD8	CD8	4.745776	Leptokurtic (heavy tails)
CD34	CD34	16.596416	Leptokurtic (heavy tails)
Flt3	Flt3	85.583534	Leptokurtic (heavy tails)
CD20	CD20	10.435449	Leptokurtic (heavy tails)
CXCR4	CXCR4	3.936307	Leptokurtic (heavy tails)
CD235ab	CD235ab	13.440586	Leptokurtic (heavy tails)
CD45	CD45	5.246770	Leptokurtic (heavy tails)
CD123	CD123	18.361217	Leptokurtic (heavy tails)
CD321	CD321	2.914593	Platykurtic (light tails)
CD14	CD14	23.062535	Leptokurtic (heavy tails)
CD33	CD33	10.967536	Leptokurtic (heavy tails)
CD47	CD47	2.943834	Platykurtic (light tails)
CD11c	CD11c	5.117156	Leptokurtic (heavy tails)
CD7	CD7	4.885115	Leptokurtic (heavy tails)
CD15	CD15	4.504387	Leptokurtic (heavy tails)
CD16	CD16	42.287749	Leptokurtic (heavy tails)

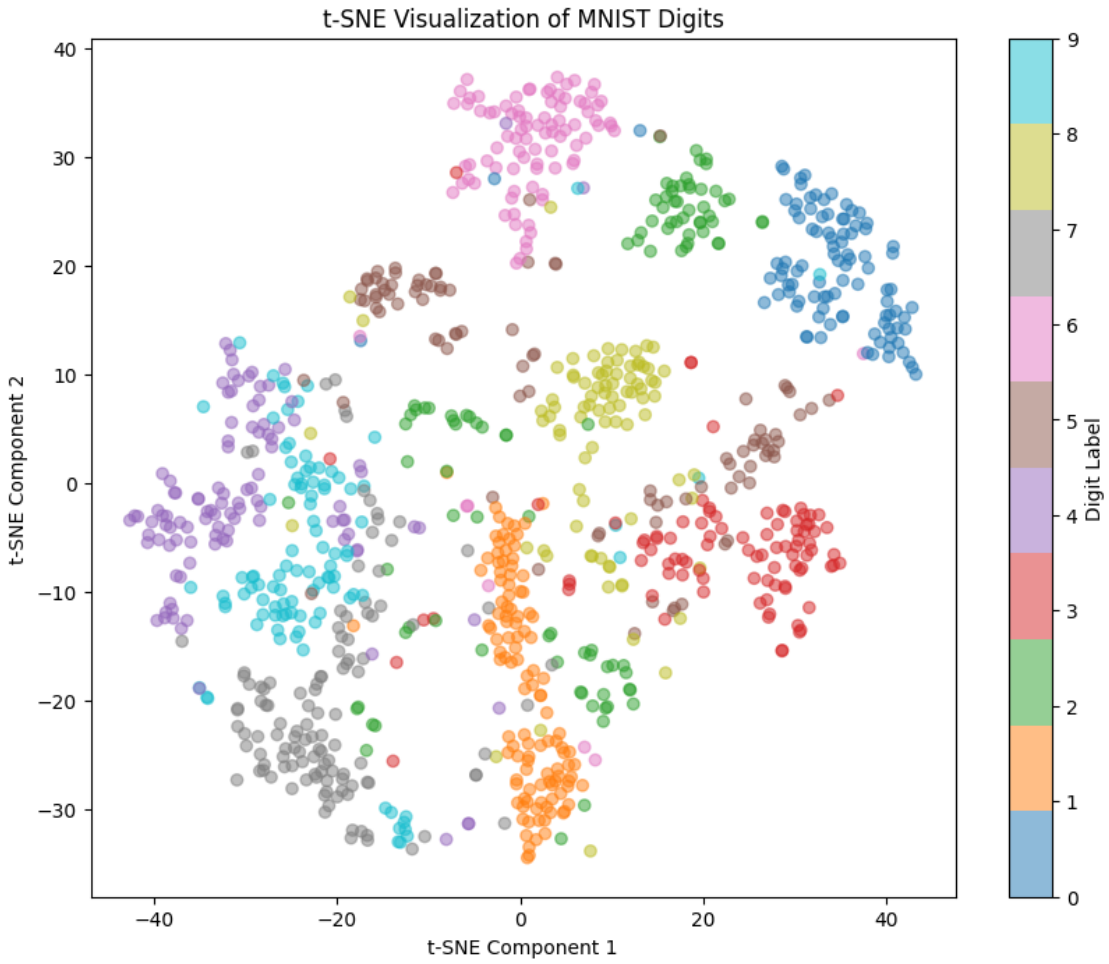
CD44	CD44	2.918792	Platykurtic (light tails)
CD38	CD38	3.521190	Leptokurtic (heavy tails)
CD13	CD13	10.637564	Leptokurtic (heavy tails)
CD3	CD3	1.264612	Platykurtic (light tails)
CD61	CD61	34.878020	Leptokurtic (heavy tails)
CD117	CD117	26.375108	Leptokurtic (heavy tails)
CD49d	CD49d	3.468119	Leptokurtic (heavy tails)
HLA-DR	HLA-DR	2.309924	Platykurtic (light tails)
CD64	CD64	4.910631	Leptokurtic (heavy tails)
CD41	CD41	41.521113	Leptokurtic (heavy tails)
Viability	Viability	3.156935	Leptokurtic (heavy tails)
event_number	event_number	1.706183	Platykurtic (light tails)
label	label	NaN	Mesokurtic (normal tails)
individual	individual	1.964382	Platykurtic (light tails)


Cell_length (Kurtosis: 2.83)
DNA1 (Kurtosis: 1.99)
DNA2 (Kurtosis: 1.98)
CD45RA (Kurtosis: 4.96)
CD133 (Kurtosis: 9.19)
CD19 (Kurtosis: 4.59)
CD22 (Kurtosis: 7.50)
CD11b (Kurtosis: 4.96)
CD4 (Kurtosis: 5.84)
CD8 (Kurtosis: 4.75)
CD34 (Kurtosis: 16.60)
Flt3 (Kurtosis: 85.58)
CD20 (Kurtosis: 10.44)
CXCR4 (Kurtosis: 3.94)
CD235ab (Kurtosis: 13.44)
CD45 (Kurtosis: 5.25)
CD123 (Kurtosis: 18.36)
CD321 (Kurtosis: 2.91)
CD14 (Kurtosis: 23.06)
CD33 (Kurtosis: 10.97)
CD47 (Kurtosis: 2.94)
CD11c (Kurtosis: 5.12)
CD7 (Kurtosis: 4.89)
CD15 (Kurtosis: 4.50)
CD16 (Kurtosis: 42.29)
CD44 (Kurtosis: 2.92)
CD38 (Kurtosis: 3.52)
CD13 (Kurtosis: 10.64)
CD3 (Kurtosis: 1.26)
CD61 (Kurtosis: 34.88)
CD117 (Kurtosis: 26.38)
CD49d (Kurtosis: 3.47)
HLA-DR (Kurtosis: 2.31)
CD64 (Kurtosis: 4.91)
CD41 (Kurtosis: 41.52)
Viability (Kurtosis: 3.16)
event_number (Kurtosis: 1.71)
label (Kurtosis: nan)
individual (Kurtosis: 1.96)
Density
Cell_length
DNA1
DNA2
CD45RA
CD133
CD19
CD22
CD11b
CD4
CD8
CD34
Flt3
CD20
CXCR4
CD235ab
CD45
CD123
CD321
CD14
CD33
CD47
CD11c
CD7
CD15
CD16
CD44
CD38
CD13
CD3
CD61
CD117
CD49d
HLA-DR
CD64
CD41
Viability
1e−6
event_number
label
individual

##T-SNE Visualization

[ ]:
```
import tensorflow as tf
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import numpy as np

# Load the MNIST dataset
(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.
 ↪mnist.load_data()
train_images = train_images.astype('float32') / 255.0
test_images = test_images.astype('float32') / 255.0

# Flatten the images and take a subset
n_samples = 1000
train_images_flat = train_images[:n_samples].reshape(n_samples, -1)
train_labels_subset = train_labels[:n_samples]

# Perform t-SNE
tsne = TSNE(n_components=2, random_state=42, perplexity=30)
train_images_embedded = tsne.fit_transform(train_images_flat)

# Plot the t-SNE results
plt.figure(figsize=(10, 8))
scatter = plt.scatter(train_images_embedded[:, 0], train_images_embedded[:, 1],␣
 ↪c=train_labels_subset, cmap='tab10', alpha=0.5)
plt.colorbar(scatter, label='Digit Label')
plt.title('t-SNE Visualization of MNIST Digits')
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.show()
```

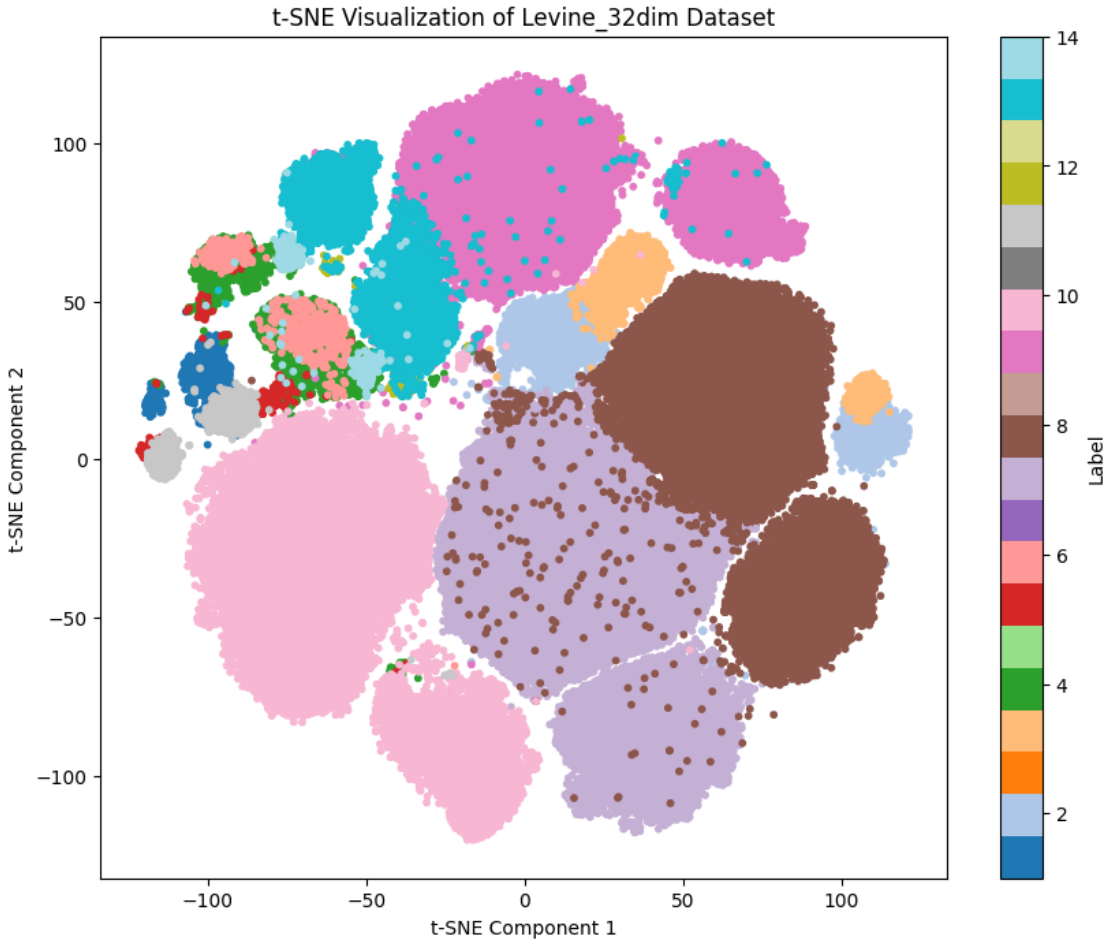
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11490434/11490434             0s
0us/step
```



##t-SNE Visualization for Dimensionality Reduction

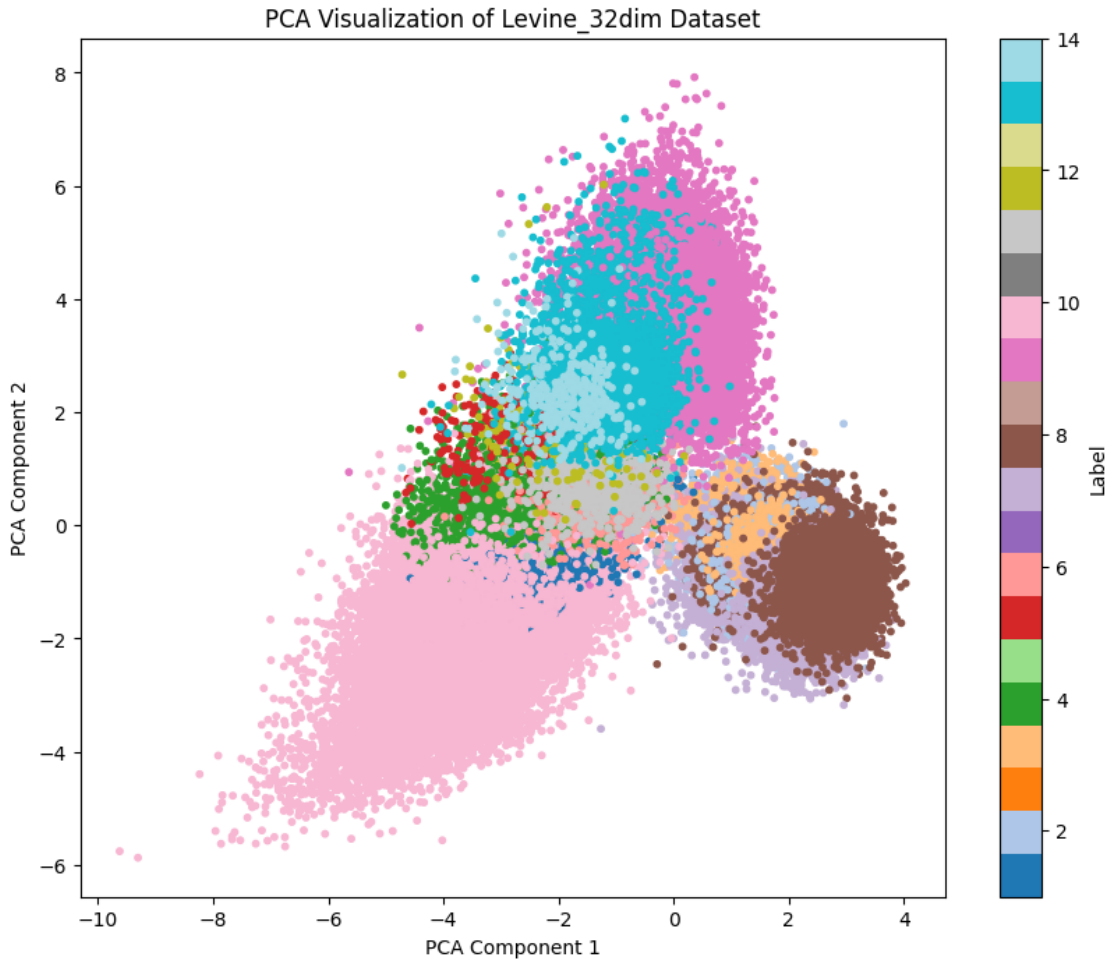
[ ]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt

# Load the dataset
# data = pd.read_csv('/content/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Standardize the data (z-score normalization)
scaler = StandardScaler()
```

```
data_standardized = scaler.fit_transform(data_filtered)

# Perform t-SNE
tsne = TSNE(n_components=2, random_state=42, perplexity=30)  # You can adjust␣
 ↪perplexity as needed
tsne_results = tsne.fit_transform(data_standardized)

# Add the t-SNE results to the original data for visualization
data['t-SNE Component 1'] = tsne_results[:, 0]
data['t-SNE Component 2'] = tsne_results[:, 1]

# Plot the t-SNE visualization
plt.figure(figsize=(10, 8))
scatter = plt.scatter(data['t-SNE Component 1'], data['t-SNE Component 2'],␣
 ↪c=data['label'], cmap='tab20', s=10)
plt.colorbar(scatter, label='Label')
plt.title('t-SNE Visualization of Levine_32dim Dataset')
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.show()
```



##Principal Component Analysis (PCA) for Dimensionality Reduction

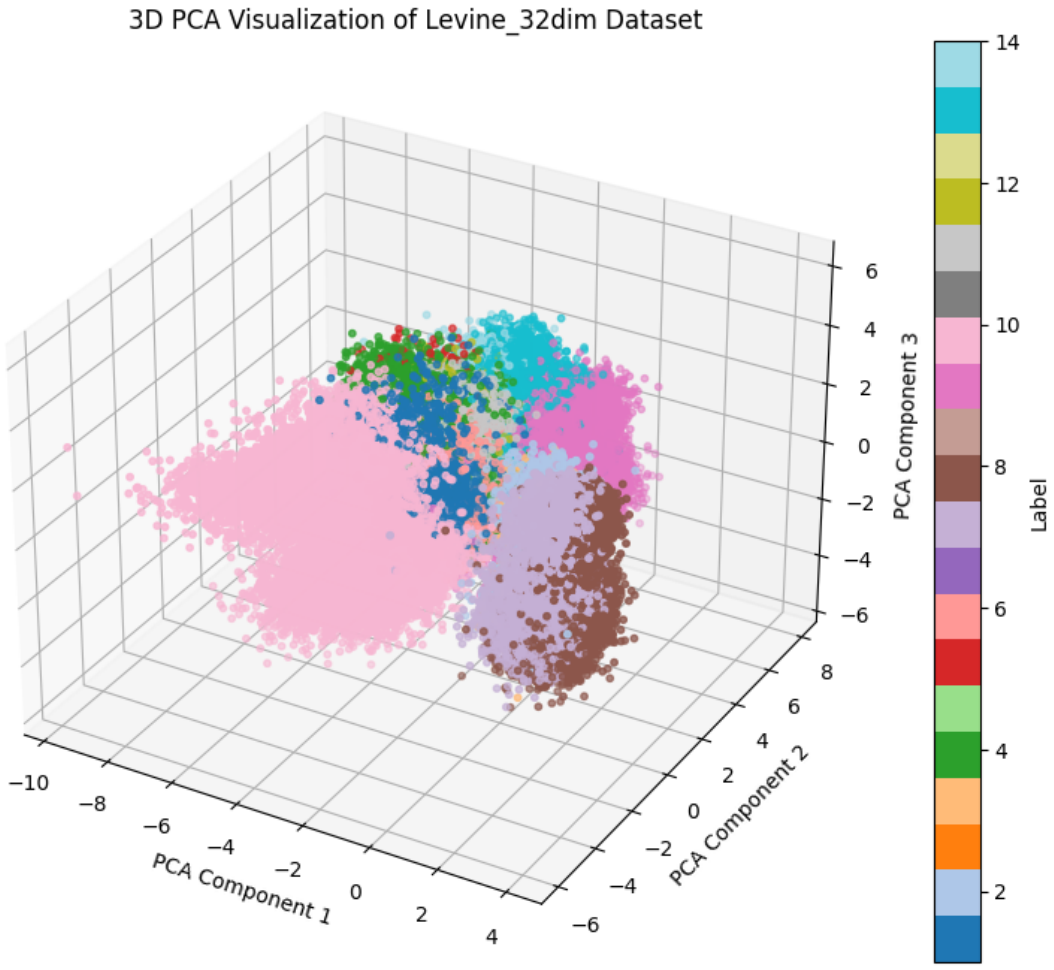
[ ]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Load the dataset
# data = pd.read_csv('/content/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Standardize the data (z-score normalization)
scaler = StandardScaler()
data_standardized = scaler.fit_transform(data_filtered)

# Perform PCA
pca = PCA(n_components=2)  # Reduce to 2 dimensions for visualization
pca_result = pca.fit_transform(data_standardized)

# Add the PCA results to the original data for visualization
data['PCA Component 1'] = pca_result[:, 0]
data['PCA Component 2'] = pca_result[:, 1]

# Plot the PCA results
plt.figure(figsize=(10, 8))
scatter = plt.scatter(data['PCA Component 1'], data['PCA Component 2'],␣
 ↪c=data['label'], cmap='tab20', s=10)
plt.colorbar(scatter, label='Label')
plt.title('PCA Visualization of Levine_32dim Dataset')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.show()
```



##3D PCA graph

[ ]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D  # Importing 3D plotting

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Standardize the data (z-score normalization)
```

```
scaler = StandardScaler()
data_standardized = scaler.fit_transform(data_filtered)

# Perform PCA
pca = PCA(n_components=3)  # Reduce to 3 dimensions for 3D visualization
pca_result = pca.fit_transform(data_standardized)

# Add the PCA results to the original data for visualization
data['PCA Component 1'] = pca_result[:, 0]
data['PCA Component 2'] = pca_result[:, 1]
data['PCA Component 3'] = pca_result[:, 2]

# Plot the PCA results in 3D
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

# Create a 3D scatter plot
scatter = ax.scatter(data['PCA Component 1'], data['PCA Component 2'],␣
 ↪data['PCA Component 3'],
                     c=data['label'], cmap='tab20', s=10)

# Add color bar and labels
plt.colorbar(scatter, label='Label')
ax.set_title('3D PCA Visualization of Levine_32dim Dataset')
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_zlabel('PCA Component 3')

# Show the plot
plt.show()
```



##Variance, Cumulative Proportion, and Standard Deviation Analysis

[ ]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Standardize the data (z-score normalization)
scaler = StandardScaler()
data_standardized = scaler.fit_transform(data_filtered)
```

```
# Perform PCA
pca = PCA(n_components=4)  # Use 4 principal components
pca.fit(data_standardized)

# Extract the required information
explained_variance = pca.explained_variance_ratio_
cumulative_variance = explained_variance.cumsum()
standard_deviation = pca.singular_values_ / (len(data_standardized) - 1)**0.5

# Create a DataFrame for the output
pca_summary = pd.DataFrame({
    'PC1': [standard_deviation[0], explained_variance[0],␣
 ↪cumulative_variance[0]],
    'PC2': [standard_deviation[1], explained_variance[1],␣
 ↪cumulative_variance[1]],
    'PC3': [standard_deviation[2], explained_variance[2],␣
 ↪cumulative_variance[2]],
    'PC4': [standard_deviation[3], explained_variance[3],␣
 ↪cumulative_variance[3]]
}, index=['Standard Deviation', 'Proportion of Variance', 'Cumulative␣
 ↪Proportion'])

# Round the numbers for better readability
pca_summary = pca_summary.map(lambda x: f'{x:.4f}')

# Apply styles to the DataFrame
styled_summary = (pca_summary.style
                  .set_caption("PCA Summary")
                  .set_table_styles(
                      [{'selector': 'caption', 'props': [('font-size', '16px'),␣
 ↪('color', 'black'), ('font-weight', 'bold')]}]
                  )
                  .background_gradient(cmap='coolwarm', axis=None)
                  .set_properties(**{'text-align': 'center'})
)

# Hiding the index column manually (workaround)
styled_summary.set_table_styles({
    'index': [{'selector': '', 'props': 'display:none;'}]  # Hides the index␣
 ↪column
})

# Display the styled DataFrame
styled_summary
```

[ ]: <pandas.io.formats.style.Styler at 0x782a6f557880>

##Binary Masking

[ ]:
```
import numpy as np
import pandas as pd

# Set a random seed for reproducibility
np.random.seed(42)

# Create a sample DataFrame called 'demodata' for demonstration
demodata = pd.DataFrame({
    'column1': [5, 12, 18, 7],
    'column2': [10, 20, 15, 30],
    'column3': [25, 35, 40, 45]
})

# Define the probability of masking (e.g., 0.3 means a 30% chance each element␣
 ↪will be masked)
p_m = 0.3

# Convert 'demodata' to a NumPy array for masking
data_array = demodata.values

# Generate a binary mask based on the probability, where 1 = not masked, 0 =␣
 ↪masked
mask = np.random.binomial(1, 1 - p_m, data_array.shape)  # Reverse probability␣
 ↪for desired 1/0 output

# Convert to a DataFrame for easier analysis
binary_mask_df = pd.DataFrame(mask, columns=demodata.columns)

print("Original DataFrame:\n", demodata)
print("\nBinary Mask DataFrame:\n", binary_mask_df)
```

```
Original DataFrame:
    column1  column2  column3
0        5       10       25
1       12       20       35
2       18       15       40
3        7       30       45

Binary Mask DataFrame:
    column1  column2  column3
0        1        0        0
1        1        1        1
2        1        0        1
3        0        1        0
```

##**Random Shuffling of Data**

[ ]:
```
import numpy as np
import pandas as pd

# Create a sample DataFrame called 'demodata' for demonstration
demodata = pd.DataFrame({
    'column1': [5, 12, 18, 7],
    'column2': [10, 20, 15, 30],
    'column3': [25, 35, 40, 45]
})

# Shuffle each column in the DataFrame independently
shuffled_demodata = demodata.apply(lambda col: np.random.permutation(col))

print("Original DataFrame:\n", demodata)
print("\nShuffled DataFrame:\n", shuffled_demodata)
```

```
Original DataFrame:
    column1  column2  column3
0         5       10       25
1        12       20       35
2        18       15       40
3         7       30       45

Shuffled DataFrame:
    column1  column2  column3
0         5       30       25
1        18       15       40
2        12       10       35
3         7       20       45
```

##**Corrupted DataFrame** Formula = (x.values * (1 - m) + x_shuffled.values * m)

[ ]:
```
import numpy as np
import pandas as pd

# Create a sample DataFrame called 'x' (original data)
x = pd.DataFrame({
    'column1': [5, 12, 18, 7],
    'column2': [10, 20, 15, 30],
    'column3': [25, 35, 40, 45]
})

# Define the probability of masking (e.g., 0.3 means a 30% chance each element␣
 ↪will be masked)
p_m = 0.3
```

```
# Generate a binary mask matrix 'm'
m = np.random.binomial(1, 1 - p_m, x.shape)
binary_mask_df = pd.DataFrame(m, columns=x.columns)

# Shuffle each column in 'x' independently to create 'x_shuffled'
x_shuffled = x.apply(lambda col: np.random.permutation(col))

# Calculate the corrupted DataFrame 'x_corrupted' using the formula
x_corrupted_array = x.values * (1 - m) + x_shuffled.values * m
x_corrupted = pd.DataFrame(x_corrupted_array, columns=x.columns)

# Display results
print("Original DataFrame (x):\n", x)
print("\nBinary Mask DataFrame (m):\n", binary_mask_df)
print("\nShuffled DataFrame (x_shuffled):\n", x_shuffled)
print("\nCorrupted DataFrame (x_corrupted):\n", x_corrupted)
```

```
Original DataFrame (x):
    column1  column2  column3
0         5       10       25
1        12       20       35
2        18       15       40
3         7       30       45

Binary Mask DataFrame (m):
    column1  column2  column3
0         1        1        1
1         1        1        0
2         1        1        1
3         1        0        1

Shuffled DataFrame (x_shuffled):
    column1  column2  column3
0        12       20       40
1         7       15       45
2        18       30       25
3         5       10       35

Corrupted DataFrame (x_corrupted):
    column1  column2  column3
0        12       20       40
1         7       15       35
2        18       30       25
3         5       30       35
```

##Applying Binary Mask, Shuffling, and Handling Corrupted Data on the Original Dataset

[ ]:
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Set the probability of masking
p_m = 0.3

# Generate a binary mask matrix 'm'
m = np.random.binomial(1, 1 - p_m, data_filtered.shape)
binary_mask_df = pd.DataFrame(m, columns=data_filtered.columns)

# Shuffle each column in 'data_filtered' independently to create 'data_shuffled'
data_shuffled = data_filtered.apply(lambda col: np.random.permutation(col))

# Calculate the corrupted DataFrame 'data_corrupted' using the formula
data_corrupted_array = data_filtered.values * (1 - m) + data_shuffled.values * m
data_corrupted = pd.DataFrame(data_corrupted_array, columns=data_filtered.
 ↪columns)

# Display results
print("Binary Mask DataFrame (m):\n", binary_mask_df)
print("\nShuffled DataFrame (data_shuffled):\n", data_shuffled)
print("\nCorrupted DataFrame (data_corrupted):\n", data_corrupted)
```

```
Binary Mask DataFrame (m):
         DNA1  DNA2  CD45RA  CD133  CD19  CD22  CD11b  CD4  CD8  CD34  …  \
0           1     1       1      1     1     1      1    0    1     1  …
1           1     1       1      1     1     0      0    1    0     1  …
2           1     1       1      1     0     1      1    0    1     1  …
3           1     1       1      0     1     0      1    0    1     1  …
4           1     0       1      1     0     1      0    1    1     1  …
…       …   …     …       …    …     …     …    …     …    …     …  …
265622      0     1       0      1     0     1      1    1    1     1  …
265623      1     0       1      0     0     0      1    0    0     1  …
265624      0     1       1      1     1     1      0    1    1     0  …
265625      1     1       1      1     1     1      1    0    0     1  …
265626      0     1       1      1     1     1      0    1    0     1  …

        CD38  CD13  CD3  CD61  CD117  CD49d  HLA-DR  CD64  CD41  Viability
```

```
0          0     0    1     1      0      1       1     1     1          1
1          1     1    0     1      1      1       0     1     1          1
2          1     1    1     1      1      1       1     1     1          1
3          0     1    1     1      0      1       1     0     0          1
4          0     0    0     0      1      1       1     0     0          1
…      …   …   …   …    …     …     …      …   …   …        …
265622     1     1    1     1      1      1       1     1     1          1
265623     1     1    1     1      1      0       0     1     1          1
265624     1     0    1     0      1      1       1     1     0          1
265625     1     1    1     0      1      1       0     1     1          1
265626     0     1    1     1      0      0       1     0     1          1

[265627 rows x 35 columns]

Shuffled DataFrame (data_shuffled):
             DNA1      DNA2    CD45RA     CD133      CD19      CD22     CD11b  \
0        3.493646  6.465404  0.340999 -0.025332 -0.025103  0.324469  0.232285
1        3.114533  3.423132  0.213500  0.126525  0.146661  0.173498  0.682279
2        3.836141  4.463463  0.060788  0.096401  0.732043 -0.015207  1.114383
3        3.581499  4.787491  0.359208  0.908222 -0.001697  1.677991 -0.034786
4        3.610631  6.638976  0.286461  0.217847  1.917009  3.022127  0.565762
…             …       …        …         …         …         …         …
265622   6.743548  4.322536 -0.005705 -0.016268  0.582740 -0.019210  0.828056
265623   3.347143  4.877782 -0.004178 -0.034062 -0.039304 -0.007548  3.378285
265624   3.686621  3.506769  0.498960  0.271947  2.187979 -0.037916 -0.042128
265625   4.187246  4.474399  0.601194 -0.042603 -0.023895 -0.010985  0.795082
265626   3.921330  6.933016  0.055985 -0.009332  0.100239  0.924512 -0.000688

              CD4       CD8      CD34  …      CD38      CD13       CD3  \
0        1.482412  0.089950 -0.006378  …  1.248795 -0.046509  4.787287
1        0.152495  0.208741  0.447693  …  2.460383  2.510161  4.297918
2       -0.033860  0.956316  0.044209  …  0.086228  0.702156 -0.013035
3       -0.028634 -0.043504  0.148585  …  0.625134  0.078449  0.061610
4        0.959473  0.142730  0.788850  …  0.164437  0.578504  4.080025
…             …       …        …  …       …         …         …
265622   1.838663  0.003775  0.066474  …  4.302753  0.656423  5.322266
265623   0.072676 -0.036365  0.071084  …  1.932774  0.048285  5.147650
265624  -0.025360 -0.009409  0.704274  …  0.306596  0.166993  3.837887
265625  -0.006787  0.280838 -0.038268  …  0.518423  0.145372 -0.006739
265626   0.907200 -0.009402  0.197751  …  1.121716  0.116836  5.221941

             CD61      CD117      CD49d     HLA-DR      CD64      CD41  Viability
0       -0.004932 -0.030021  1.035329  1.786517  0.356155 -0.023658  -0.035321
1        0.840316 -0.015361  0.098874  2.486226  0.525506  0.209810  -0.015242
2        1.646509 -0.039927 -0.040118 -0.007150  0.141350  0.057683   0.005890
3        0.075998  0.721599  0.659726  0.021337  0.086713  0.151819   0.054490
4       -0.014726  0.127639  1.480745  3.317302  0.798442  0.686276  -0.025760
…             …       …        …         …         …         …         …
```

```
265622  0.100799 -0.014003  1.588064  0.136865  3.681711 -0.041774   1.432528
265623 -0.000904  0.029533  0.365732  3.485335  1.312856  0.166331   1.293927
265624 -0.029847  0.508918  0.536416  0.639527  0.153479  0.226226  -0.045239
265625  0.437355 -0.006220  0.575870 -0.043981  0.253740  0.382460  -0.042468
265626 -0.039827 -0.008043  0.893371  0.096954  0.851119 -0.002956   1.593338

[265627 rows x 35 columns]

Corrupted DataFrame (data_corrupted):
             DNA1      DNA2    CD45RA     CD133      CD19      CD22     CD11b  \
0        3.493646  6.465404  0.340999 -0.025332 -0.025103  0.324469  0.232285
1        3.114533  3.423132  0.213500  0.126525  0.146661  0.074409  0.808031
2        3.836141  4.463463  0.060788  0.096401  0.073855 -0.015207  1.114383
3        3.581499  4.787491  0.359208 -0.027611 -0.001697 -0.044072 -0.034786
4        3.610631  4.506433  0.286461  0.217847  0.080423  3.022127  1.107627
…             …         …         …         …         …         …         …
265622   6.826629  4.322536  1.474081 -0.016268 -0.055620 -0.019210  0.828056
265623   3.347143  7.154026 -0.004178 -0.056213 -0.008864 -0.035158  3.378285
265624   6.889866  3.506769  0.498960  0.271947  2.187979 -0.037916 -0.034641
265625   4.187246  4.474399  0.601194 -0.042603 -0.023895 -0.010985  0.795082
265626   6.887820  6.933016  0.055985 -0.009332  0.100239  0.924512  3.864711

              CD4       CD8      CD34  …      CD38      CD13       CD3  \
0        0.363602  0.089950 -0.006378  …  1.395208  0.038552  4.787287
1        0.152495 -0.010551  0.447693  …  2.460383  2.510161 -0.043466
2       -0.008781  0.956316  0.044209  …  0.086228  0.702156 -0.013035
3       -0.019066 -0.043504  0.148585  …  4.147996  0.078449  0.061610
4        0.959473  0.142730  0.788850  …  3.711521  0.585712  0.137186
…             …         …         …  …         …         …         …
265622   1.838663  0.003775  0.066474  …  4.302753  0.656423  5.322266
265623   0.970120 -0.023903  0.071084  …  1.932774  0.048285  5.147650
265624  -0.025360 -0.009409  0.107905  …  0.306596  0.104754  3.837887
265625   0.078800 -0.000954 -0.038268  …  0.518423  0.145372 -0.006739
265626   0.907200  0.113039  0.197751  …  0.057280  0.116836  5.221941

              CD61      CD117      CD49d     HLA-DR       CD64       CD41  \
0       -0.004932  0.053050  1.035329  1.786517  0.356155 -0.023658
1        0.840316 -0.015361  0.098874  0.491592  0.525506  0.209810
2        1.646509 -0.039927 -0.040118 -0.007150  0.141350  0.057683
3        0.075998  0.066470  0.659726  0.021337 -0.013449 -0.026039
4        0.168609  0.127639  1.480745  3.317302  0.076167 -0.040488
…             …         …         …         …         …         …
265622   0.100799 -0.014003  1.588064  0.136865  3.681711 -0.041774
265623  -0.000904  0.029533  1.269464  0.047215  1.312856  0.166331
265624  -0.008680  0.508918  0.536416  0.639527  0.153479 -0.042602
265625  -0.029347 -0.006220  0.575870  6.200001  0.253740  0.382460
265626  -0.039827  0.080195  0.037962  0.096954 -0.000878 -0.002956

   Viability
0  -0.035321
1  -0.015242
2   0.005890
3   0.054490
4  -0.025760
…         …
265622   1.432528
265623   1.293927
265624  -0.045239
265625  -0.042468
265626   1.593338
```

[265627 rows x 35 columns]

##**New Masking** Formula = (mask_new = 1 * (data_filtered != data_corrupted))

[ ]:
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Exclude the specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Set the probability of masking
p_m = 0.3

# Generate a binary mask matrix 'm' (changes every run)
m = np.random.binomial(1, 1 - p_m, data_filtered.shape)
binary_mask_df = pd.DataFrame(m, columns=data_filtered.columns)

# Shuffle each column in 'data_filtered' independently to create␣
 ↪'data_shuffled' (changes every run)
data_shuffled = data_filtered.apply(lambda col: np.random.permutation(col))

# Calculate the corrupted DataFrame 'data_corrupted' using the formula
data_corrupted_array = data_filtered.values * (1 - m) + data_shuffled.values * m
data_corrupted = pd.DataFrame(data_corrupted_array, columns=data_filtered.
 ↪columns)

# Generate mask_new to indicate differences between original and corrupted data
mask_new = 1 * (data_filtered != data_corrupted)

# Print only the new mask matrix
print("New Mask Matrix (mask_new):\n", mask_new)
```

```
New Mask Matrix (mask_new):
         DNA1  DNA2  CD45RA  CD133  CD19  CD22  CD11b  CD4  CD8  CD34  …  \
0           1     1       1      1     1     0      0    0    1     1  …
1           1     1       0      1     0     1      1    1    1     1  …
2           1     1       1      1     0     1      0    1    1     1  …
3           1     0       1      1     1     1      0    0    0     1  …
4           1     1       0      1     0     0      1    1    1     1  …
…       …   …     …       …    …     …     …    …     …    …    …     …  …
265622      1     1       1      1     0     0      1    0    0     1  …
265623      0     0       1      1     0     1      1    1    1     1  …
```

```
265624      1     1       1      1     1     0      1    1    1     1  ...
265625      1     1       0      0     1     0      1    1    0     1  ...
265626      0     0       0      1     1     1      1    1    0     1  ...

        CD38  CD13  CD3  CD61  CD117  CD49d  HLA-DR  CD64  CD41  Viability
0          1     1    1     1      1      1       1     1     1          1
1          1     1    1     1      0      0       0     0     0          1
2          0     1    1     1      0      1       0     1     0          1
3          1     0    1     1      1      1       1     0     1          0
4          1     1    1     0      1      1       0     1     1          1
...    ...   ...  ...  ...   ...    ...    ...   ...  ...     ...
265622     1     1    1     1      1      0       1     1     1          1
265623     1     1    1     1      1      1       0     0     1          0
265624     1     1    1     1      1      0       0     1     1          1
265625     1     0    1     1      1      0       1     1     1          0
265626     0     0    1     1      1      1       1     0     0          1

[265627 rows x 35 columns]
```

##**Separating Features and Labels in Unlabeled Data**

[ ]:

```
import numpy as np
import pandas as pd

# Load the dataset
# df = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Define the target column used for labeling
label_column = 'label'

# Separate labeled and unlabeled data using label_df
label_df = df[df[label_column].notnull()]  # labeled data
unlabeled_df = df[df[label_column].isnull()]  # unlabeled data

# Split features and labels for labeled data
x_labeled = label_df.drop(columns=[label_column])
y_labeled = label_df[label_column]

# Split features and labels for unlabeled data
x_unlabeled = unlabeled_df.drop(columns=[label_column])
y_unlabeled = unlabeled_df[label_column]

# Display results
print("Labeled Features (x_labeled):\n", x_labeled)
print("\nLabeled Labels (y_labeled):\n", y_labeled)
print("\nUnlabeled Features (x_unlabeled):\n", x_unlabeled)
print("\nUnlabeled Labels (y_unlabeled):\n", y_unlabeled)
```

Labeled Features (x_labeled):

	Event	Time	Cell_length	DNA1	DNA2	CD45RA	\
0	1	2693.00	22	4.391057	4.617262	0.162691	
1	2	3736.00	35	4.340481	4.816692	0.701349	
2	3	7015.00	32	3.838727	4.386369	0.603568	
3	4	7099.00	29	4.255806	4.830048	0.433747	
4	5	7700.00	25	3.976909	4.506433	-0.008809	
…	…	…	…	…	…	…	
104179	104180	641812.44	58	6.827981	7.249403	-0.000106	
104180	104181	653387.44	55	6.683204	7.166172	0.692668	
104181	104182	671024.44	40	6.911546	7.152603	-0.036795	
104182	104183	680006.44	48	6.700332	7.100771	0.308817	
104183	104184	687494.44	64	6.559460	7.080928	0.519572	

	CD133	CD19	CD22	CD11b	…	CD61	CD117	\
0	-0.029585	-0.006696	0.066388	-0.009184	…	-0.002936	0.053050	
1	-0.038280	-0.016654	0.074409	0.808031	…	1.258437	0.089660	
2	-0.032216	0.073855	-0.042977	-0.001881	…	0.257137	0.046222	
3	-0.027611	-0.017661	-0.044072	0.733698	…	-0.041140	0.066470	
4	-0.030297	0.080423	0.495791	1.107627	…	0.168609	-0.006223	
…	…	…	…	…	…	…		
104179	-0.030641	1.432347	-0.044946	-0.016534	…	0.188846	-0.002144	
104180	-0.037335	1.639063	0.286325	-0.036985	…	-0.029213	-0.031301	
104181	-0.014477	1.637975	-0.021794	-0.020169	…	-0.015220	-0.034755	
104182	0.075762	1.455129	0.042576	-0.049737	…	-0.016644	-0.047522	
104183	0.097257	1.346523	0.279473	-0.021585	…	-0.051973	-0.017015	

	CD49d	HLA-DR	CD64	CD41	Viability	file_number	\
0	0.853505	1.664480	-0.005376	-0.001961	0.648429	3.627711	
1	0.197818	0.491592	0.144814	0.868014	0.561384	3.627711	
2	2.586670	1.308337	-0.010961	-0.010413	0.643337	3.627711	
3	1.338669	0.140523	-0.013449	-0.026039	-0.026523	3.627711	
4	0.180924	0.197332	0.076167	-0.040488	0.283287	3.627711	
…	…	…	…	…	…	…	
104179	1.115652	2.373524	-0.004620	-0.051592	0.157816	3.669327	
104180	1.653418	4.367032	0.062683	0.158656	0.025255	3.669327	
104181	1.083173	3.541526	0.110382	0.108349	-0.043739	3.669327	
104182	0.432565	3.882030	0.058852	0.185295	0.204898	3.669327	
104183	0.263008	4.332834	-0.017214	0.130106	0.023135	3.669327	

	event_number	individual
0	307	1
1	545	1
2	1726	1
3	1766	1
4	2031	1
…	…	…
104179	100344	2

```
104180        100892           2
104181        101558           2
104182        101842           2
104183        102112           2

[104184 rows x 41 columns]

Labeled Labels (y_labeled):
 0          1.0
1           1.0
2           1.0
3           1.0
4           1.0
           …
104179     14.0
104180     14.0
104181     14.0
104182     14.0
104183     14.0
Name: label, Length: 104184, dtype: float64

Unlabeled Features (x_unlabeled):
          Event       Time  Cell_length      DNA1      DNA2    CD45RA  \
104184  104185      40.00           25  4.203073  4.837565  0.095543
104185  104186     176.00           34  4.042991  4.808275  0.035310
104186  104187     189.00           37  4.233125  4.922201  0.415954
104187  104188     193.00           26  3.997143  4.685426 -0.038565
104188  104189     204.00           20  4.115830  4.893428  0.177246
…         …         …           …        …       …       …
265622  265623  707951.44           41  6.826629  7.133022  1.474081
265623  265624  708145.44           45  6.787791  7.154026  0.116755
265624  265625  708398.44           41  6.889866  7.141219  0.684921
265625  265626  708585.44           39  6.865218  7.144353  0.288761
265626  265627  709122.44           41  6.887820  7.127359  0.360753

           CD133      CD19      CD22     CD11b  …      CD61     CD117  \
104184 -0.027206  0.172384 -0.001950  0.505713  …  3.029787 -0.010093
104185 -0.013869 -0.043922 -0.001871  0.180261  … -0.017628  0.346248
104186  0.412757  0.431715 -0.025619  0.491190  …  0.000544  0.691393
104187  0.125894  0.191383 -0.026497  0.342190  … -0.012887  0.033096
104188  0.171916  0.028568 -0.029751  2.480689  … -0.015719 -0.043689
…          …         …        …        …  …        …        …
265622 -0.019174 -0.055620 -0.007261  0.063395  …  0.861068 -0.011105
265623 -0.056213 -0.008864 -0.035158 -0.041845  …  0.565170  0.143869
265624 -0.006264 -0.026111 -0.030837 -0.034641  … -0.008680  0.087102
265625 -0.011310 -0.048786  0.073983 -0.031787  … -0.029347 -0.047971
265626  0.128604 -0.006934  0.109846  3.864711  … -0.023831  0.080195
```

```
           CD49d    HLA-DR      CD64      CD41  Viability  file_number  \
104184  0.387121  2.859639  2.709532  1.208795   0.102978     3.627711
104185  0.089940 -0.017702  0.045091 -0.022009   0.092770     3.627711
104186  2.996583  5.812406  1.713608  0.479122   1.888485     3.627711
104187 -0.029722 -0.031126 -0.020739 -0.014693   0.067437     3.627711
104188  0.027586  2.543139  3.323810 -0.002918   0.109243     3.627711
…            …         …         …         …          …
265622  0.533736  0.123758 -0.042495 -0.027971   0.236957     3.669327
265623  1.269464  0.047215 -0.008000 -0.025811  -0.003500     3.669327
265624 -0.055912  0.501536  0.053884 -0.042602   0.107206     3.669327
265625  0.101955  6.200001  0.296877  0.192786   0.620872     3.669327
265626  0.037962  3.675123 -0.000878 -0.052526   0.310466     3.669327

        event_number  individual
104184             1           1
104185             6           1
104186             7           1
104187             8           1
104188             9           1
…                …           …
265622        102686           2
265623        102690           2
265624        102701           2
265625        102706           2
265626        102720           2

[161443 rows x 41 columns]

Unlabeled Labels (y_unlabeled):
 104184   NaN
104185   NaN
104186   NaN
104187   NaN
104188   NaN
          ..
265622   NaN
265623   NaN
265624   NaN
265625   NaN
265626   NaN
Name: label, Length: 161443, dtype: float64
```

##Splitting Labeled Dataset into Training and Testing Sets (70% Training, 30% Testing)

[ ]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
# Load the dataset
# df = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Define the target column used for labeling
label_column = 'label'

# Separate labeled data
label_df = df[df[label_column].notnull()]

# Split features and labels for labeled data
x_labeled = label_df.drop(columns=[label_column])
y_labeled = label_df[label_column]

# Split labeled data into training and testing sets (70%-30% split)
x_train, x_test, y_train, y_test = train_test_split(x_labeled, y_labeled,␣
 ↪test_size=0.3, random_state=42)

# Display results
print("Training Features (x_train):\n", x_train)
print("\nTesting Features (x_test):\n", x_test)
print("\nTraining Labels (y_train):\n", y_train)
print("\nTesting Labels (y_test):\n", y_test)
```

```
Training Features (x_train):
         Event       Time  Cell_length      DNA1      DNA2    CD45RA  \
64113    64114  401196.00           25  3.899656  4.594272  0.976652
82744    82745  502826.44           31  6.592998  6.901888  0.431481
24294    24295  488377.00           41  3.543583  4.467671  0.377192
7820      7821  225689.00           38  4.305227  4.881685  0.199351
43295    43296  153333.00           26  4.159271  4.861015  0.831285
…          …          …          …         …         …         …
54886    54887   93991.00           15  4.074604  4.747052  0.431805
76820    76821   46189.00           33  6.584427  6.882117  0.640424
103694  103695  574005.44           43  6.719895  7.080995  0.306443
860        861  516979.00           26  3.886782  4.886936  0.060176
15795    15796  225860.00           25  3.523293  4.289820  0.646288

           CD133      CD19      CD22     CD11b  …      CD61     CD117  \
64113   0.302811  0.154761 -0.011676  3.180236  …  0.051464 -0.003680
82744  -0.052898 -0.037690 -0.029715 -0.040846  … -0.036430  0.021689
24294   0.219081  0.245478  0.193328  0.075123  …  1.003383  0.406137
7820    0.100678 -0.025812 -0.002898  1.437247  … -0.007282  1.421540
43295   0.191518  2.002712  3.387782  0.179219  … -0.040754  0.060944
…            …         …         …         …  …         …         …
54886   0.228761 -0.011434 -0.017082  1.379518  … -0.029607 -0.039425
76820  -0.044057 -0.013737 -0.030704 -0.009781  … -0.038000  0.190509
```

```
103694 -0.026339  2.074008  0.052549  0.167479  …  0.054690  0.011329
860     0.233401 -0.020592 -0.007786  1.090780  … -0.001868 -0.046200
15795  -0.028126  0.184879  0.214664  0.224471  …  0.089666  0.343049

           CD49d    HLA-DR      CD64      CD41  Viability  file_number  \
64113   1.260410  0.700093  2.355886  0.125409   0.840205     3.627711
82744   0.034946 -0.055651 -0.023248 -0.054842  -0.009329     3.669327
24294   1.928676 -0.046849  0.229309  0.937020   1.231347     3.627711
7820    1.443145  2.461705  0.528679  0.072205   0.892480     3.627711
43295   1.294561  3.085858 -0.014128  0.479256   2.269233     3.627711
…          …       …        …         …          …           …
54886   0.036619  2.424191  1.080756 -0.014481   0.190138     3.627711
76820   0.204920 -0.004600  0.135288 -0.042874  -0.023160     3.669327
103694  0.267845  4.060155  0.123218  0.006991  -0.026324     3.669327
860     1.016980  0.000744 -0.030356 -0.033473   0.371143     3.627711
15795   0.784416  0.064465  0.088172 -0.013586   0.153918     3.627711

        event_number  individual
64113         318320           1
82744          80934           2
24294         366690           1
7820          203131           1
43295         152117           1
…                …          …
54886          96894           1
76820           8563           2
103694         94148           2
860           378748           1
15795         203230           1

[72928 rows x 41 columns]

Testing Features (x_test):
        Event      Time  Cell_length      DNA1      DNA2    CD45RA     CD133  \
60544   60545  278003.0           49  3.618797  4.144135  0.198186  0.000282
50673   50674  490341.0           27  3.660988  4.497041  1.272625  0.129642
50682   50683  490912.0           23  3.854865  4.663734  1.527763  0.151383
1761     1762  170466.0           17  3.716473  4.465312  0.375236 -0.037150
98760   98761  423490.0           32  6.826030  7.007709  0.223441 -0.048813
…         …       …          …        …         …         …         …
20510   20511  370777.0           63  3.260559  3.934633  0.448954  0.219533
11540   11541   99635.0           37  3.204839  3.422136  0.088893  0.359100
30042   30043  145367.0           57  3.351777  4.185945  1.148632  0.383412
40569   40570   45221.0           50  4.010990  4.529642  1.211406  1.121462
93618   93619  289293.0           37  6.732461  6.913152  1.734362  0.126751

           CD19      CD22     CD11b  …      CD61     CD117     CD49d  \
60544   0.253703 -0.018972  2.665005  …  0.307357  0.208639  2.039954
```

```
50673  3.054480  2.493220  0.189975  …  0.084448  0.033192  0.004637
50682  2.361353  2.281009  0.528589  … -0.041903 -0.026017  0.109363
1761  -0.035385  0.127904  0.415204  … -0.001024 -0.017034  0.023385
98760 -0.018816 -0.045954  4.067125  … -0.029816 -0.046020  0.140410
…         …         …        …  …         …         …
20510  0.105799  0.093621 -0.006647  …  0.599577  0.376384  2.196247
11540 -0.001227  0.128556  0.008345  …  0.908547  0.001992  0.464461
30042 -0.037390  0.229479  0.005238  …  0.596622  0.055177  0.761682
40569  1.185200  0.905587  0.254603  …  0.120182 -0.007947  1.649371
93618  1.406384  1.672294  0.082506  … -0.033528 -0.011614  0.134475

         HLA-DR      CD64      CD41  Viability  file_number  event_number  \
60544  2.847283  2.798986  1.090235   1.005784     3.627711        237532
50673  4.488360  0.866820 -0.002174   0.917810     3.627711        367731
50682  2.328828 -0.008223 -0.018680   1.091297     3.627711        367970
1761   0.120367  0.472159 -0.014919   0.620643     3.627711        164637
98760  0.735830  1.011186 -0.044875   0.149759     3.669327         62492
…         …         …        …         …           …           …
20510  0.342656  0.235691  0.128557   1.251073     3.627711        298390
11540 -0.011717  0.331829  0.804992   1.791590     3.627711        103618
30042  0.194395  0.496897  1.122718   0.614461     3.627711        146117
40569  3.598308  0.521024  0.592218   1.099637     3.627711         37211
93618  1.677873  0.355002 -0.013528  -0.017024     3.669327         56333

       individual
60544           1
50673           1
50682           1
1761            1
98760           2
…             …
20510           1
11540           1
30042           1
40569           1
93618           2

[31256 rows x 41 columns]

Training Labels (y_train):
 64113     10.0
82744      7.0
24294      7.0
7820       6.0
43295      9.0
         …
54886     10.0
76820      7.0
```

```
103694    13.0
860        1.0
15795      7.0
Name: label, Length: 72928, dtype: float64

Testing Labels (y_test):
 60544    10.0
50673     9.0
50682     9.0
1761      2.0
98760    10.0
         …
20510     7.0
11540     7.0
30042     8.0
40569     9.0
93618     9.0
Name: label, Length: 31256, dtype: float64
```

##**Logistic Regression and XGBoost Models**

[ ]:
```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Load the dataset
# df = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Define the target column used for labeling
label_column = 'label'

# Separate labeled data
label_df = df[df[label_column].notnull()]

# Split features and labels for labeled data
x_labeled = label_df.drop(columns=[label_column])
y_labeled = label_df[label_column]

# Encode labels if necessary (e.g., for non-numeric labels)
label_encoder = LabelEncoder()
y_labeled = label_encoder.fit_transform(y_labeled)

# Split labeled data into training and testing sets (70%-30% split)
x_train, x_test, y_train, y_test = train_test_split(x_labeled, y_labeled,␣
 ↪test_size=0.3, random_state=42)
```

```python
# Scale features for Logistic Regression and XGBoost
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)

# Logistic Regression Model with increased max_iter and scaled data
logistic_model = LogisticRegression(max_iter=2000)
logistic_model.fit(x_train_scaled, y_train)
y_test_hat_logistic = logistic_model.predict_proba(x_test_scaled)

# XGBoost Model (using scaled data)
xgb_model = XGBClassifier(eval_metric='mlogloss')
xgb_model.fit(x_train_scaled, y_train)  # Use scaled data for training
y_test_hat_xgb = xgb_model.predict_proba(x_test_scaled)  # Use scaled test data␣
↪for prediction

# Display the predicted probabilities for Logistic Regression and XGBoost
print("Logistic Regression Predicted Probabilities:\n", y_test_hat_logistic)
print("\nXGBoost Predicted Probabilities:\n", y_test_hat_xgb)
```

```
Logistic Regression Predicted Probabilities:
 [[8.71532014e-11 7.15777512e-16 5.86413813e-12 … 3.71679134e-10
  2.57615664e-08 2.92952487e-08]
 [3.89684389e-14 5.45734530e-13 1.08967360e-13 … 9.25120234e-10
  3.73081989e-05 6.16167181e-09]
 [1.84225273e-11 5.90550517e-10 1.12355888e-11 … 1.59113271e-11
  1.62867176e-05 6.66716711e-10]
 …
 [5.05774251e-10 1.66206209e-05 5.93223532e-09 … 3.54583853e-09
  3.57005965e-11 9.95221511e-09]
 [6.94828818e-11 7.05048242e-10 2.51660858e-10 … 1.19779551e-11
  1.34185138e-06 2.54193721e-06]
 [3.32647156e-10 1.39746319e-06 5.84840147e-10 … 1.09125624e-13
  1.93385444e-05 1.73712999e-10]]

XGBoost Predicted Probabilities:
 [[8.8804103e-07 8.2875789e-07 5.7034327e-07 … 1.1134865e-06
  7.0003387e-07 9.7590839e-07]
 [7.1397778e-07 7.8185877e-07 5.3256036e-07 … 1.1057809e-06
  6.3680345e-06 1.2659862e-06]
 [8.2513844e-07 9.2658485e-07 6.4014063e-07 … 9.4035636e-07
  2.3140719e-06 1.1265030e-06]
 …
 [4.7537603e-07 1.6399291e-06 4.6528021e-07 … 5.1612403e-07
  4.0387292e-07 4.1018055e-07]
 [3.5595222e-06 3.8425301e-06 3.3242245e-06 … 4.7351091e-06
```

```
 7.2303219e-06 2.1757294e-05]
[1.9763070e-06 1.7193395e-06 1.8571778e-06 … 2.2942536e-06
 3.7291477e-06 2.6302241e-06]]
```

##Logistic Regression Log Loss

[ ]:
```
from sklearn.metrics import log_loss

# Calculate log loss for Logistic Regression
logistic_loss = log_loss(y_test, y_test_hat_logistic)
print("Logistic Regression Log Loss:", logistic_loss)
```

```
Logistic Regression Log Loss: 0.007401566937746702
```

##XGBoost Log Loss

[ ]:
```
from sklearn.metrics import log_loss


# Calculate log loss for XGBoost
xgb_loss = log_loss(y_test, y_test_hat_xgb)
print("XGBoost Log Loss:", xgb_loss)
```

```
XGBoost Log Loss: 0.0014473331046161019
```

##**ENCODER MODEL**

[ ]:
```
from keras.layers import Input, Dense
from keras.models import Model
import numpy as np

def binary_mask(p_m, data):
    """Generates a binary mask with probability p_m."""
    return np.random.binomial(1, 1 - p_m, data.shape)

def corruption(mask, data):
    num_samples, num_features = data.shape
    shuffled_data = np.zeros([num_samples, num_features])

    for feature_idx in range(num_features):
        shuffled_indices = np.random.permutation(num_samples)
        shuffled_data[:, feature_idx] = data[shuffled_indices, feature_idx]

    data_corrupted = data * (1 - mask) + shuffled_data * mask
    mask_new = (data != data_corrupted).astype(int)

    return mask_new, data_corrupted

def self_supervised(x_unlabeled, p_m, alpha, parameters):
    epochs = parameters['epochs']
```

```
    batch_size = parameters['batch_size']
    _, dimension = x_unlabeled.shape

    # Define model architecture
    input_layer = Input(shape=(dimension,))
    h = Dense(int(dimension), activation='relu')(input_layer)

    output1 = Dense(int(dimension), activation='sigmoid',␣
↪name='mask_estimation')(h)
    output2 = Dense(int(dimension), activation='sigmoid',␣
↪name='feature_estimation')(h)

    model = Model(inputs=input_layer, outputs=[output1, output2])

    # Compile model with appropriate loss functions and weights
    model.compile(
        optimizer="rmsprop",
        loss={'mask_estimation': 'binary_crossentropy', 'feature_estimation':␣
↪'mean_squared_error'},
        loss_weights={'mask_estimation': 1.0, 'feature_estimation':␣
↪float(alpha)}  # Corrected to use float
    )

    # Generate corrupted input and mask labels
    corruption_binary_mask = binary_mask(p_m, x_unlabeled)
    x_unlabeled_corrupted,mask_label = corruption(corruption_binary_mask,␣
↪x_unlabeled)

    assert x_unlabeled_corrupted.shape == mask_label.shape


    # Train model
    model.fit(x_unlabeled_corrupted, {'mask_estimation': mask_label,␣
↪'feature_estimation': x_unlabeled},
              epochs=epochs, batch_size=batch_size)

    # Display model summary (this will print the model's parameters)
    model.summary()

    # Define encoder
    name_of_layer = model.layers[1].name
    layer_output = model.get_layer(name_of_layer).output
    encoder = Model(inputs=model.input, outputs=layer_output)

    return encoder
```

[ ]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler

# Load the dataset
# data = pd.read_csv('/content/drive/MyDrive/Datasets/Levine_32dim.fcs.csv')

# Exclude specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Standardize the data
scaler = StandardScaler()
x_unlabeled_scaled = scaler.fit_transform(data_filtered)  # Now␣
 ↪x_unlabeled_scaled is defined

# Define other parameters
p_m = 0.3
alpha = 2.0
parameters = {
    'batch_size': 128,
    'epochs': 50,
}

# Run the self_supervised function with the scaled data
encoder_model = self_supervised(x_unlabeled_scaled, p_m, alpha, parameters)
```

```
Epoch 1/50
2076/2076            7s 2ms/step -
feature_estimation_loss: 0.0581 - loss: 2.1001 - mask_estimation_loss: 2.0420
Epoch 2/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.0097 - loss: 1.9883 - mask_estimation_loss: 1.9980
Epoch 3/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.0157 - loss: 1.9814 - mask_estimation_loss: 1.9971
Epoch 4/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.0349 - loss: 1.9636 - mask_estimation_loss: 1.9985
Epoch 5/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -0.0359 - loss: 1.9636 - mask_estimation_loss: 1.9995
Epoch 6/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -0.0523 - loss: 1.9425 - mask_estimation_loss: 1.9948
Epoch 7/50
```

```
2076/2076            6s 2ms/step -
feature_estimation_loss: -0.0805 - loss: 1.9190 - mask_estimation_loss: 1.9995
Epoch 8/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.0834 - loss: 1.9180 - mask_estimation_loss: 2.0014
Epoch 9/50
2076/2076            3s 2ms/step -
feature_estimation_loss: -0.0232 - loss: 1.9804 - mask_estimation_loss: 2.0037
Epoch 10/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.1739 - loss: 1.8210 - mask_estimation_loss: 1.9949
Epoch 11/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -0.3262 - loss: 1.6687 - mask_estimation_loss: 1.9949
Epoch 12/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.4777 - loss: 1.5217 - mask_estimation_loss: 1.9994
Epoch 13/50
2076/2076            3s 2ms/step -
feature_estimation_loss: -0.4262 - loss: 1.5784 - mask_estimation_loss: 2.0046
Epoch 14/50
2076/2076            6s 2ms/step -
feature_estimation_loss: -0.4602 - loss: 1.5433 - mask_estimation_loss: 2.0034
Epoch 15/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.4222 - loss: 1.5796 - mask_estimation_loss: 2.0018
Epoch 16/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.5310 - loss: 1.4679 - mask_estimation_loss: 1.9989
Epoch 17/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.7545 - loss: 1.2470 - mask_estimation_loss: 2.0015
Epoch 18/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.1250 - loss: 1.8759 - mask_estimation_loss: 2.0009
Epoch 19/50
2076/2076            3s 2ms/step -
feature_estimation_loss: -0.2789 - loss: 1.7216 - mask_estimation_loss: 2.0005
Epoch 20/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -1.4423 - loss: 0.5615 - mask_estimation_loss: 2.0039
Epoch 21/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -1.6331 - loss: 0.3649 - mask_estimation_loss: 1.9981
Epoch 22/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -1.6803 - loss: 0.3226 - mask_estimation_loss: 2.0029
Epoch 23/50
```

```
2076/2076            4s 2ms/step -
feature_estimation_loss: -1.3101 - loss: 0.6882 - mask_estimation_loss: 1.9983
Epoch 24/50
2076/2076            6s 2ms/step -
feature_estimation_loss: -1.5231 - loss: 0.4749 - mask_estimation_loss: 1.9980
Epoch 25/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -0.6642 - loss: 1.3345 - mask_estimation_loss: 1.9988
Epoch 26/50
2076/2076            6s 2ms/step -
feature_estimation_loss: -2.9714 - loss: -0.9676 - mask_estimation_loss: 2.0037
Epoch 27/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -1.5462 - loss: 0.4536 - mask_estimation_loss: 1.9998
Epoch 28/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -2.6026 - loss: -0.6064 - mask_estimation_loss: 1.9961
Epoch 29/50
2076/2076            6s 2ms/step -
feature_estimation_loss: -2.7313 - loss: -0.7323 - mask_estimation_loss: 1.9991
Epoch 30/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -3.5834 - loss: -1.5821 - mask_estimation_loss: 2.0013
Epoch 31/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -3.8846 - loss: -1.8840 - mask_estimation_loss: 2.0006
Epoch 32/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -3.3327 - loss: -1.3343 - mask_estimation_loss: 1.9984
Epoch 33/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -4.0114 - loss: -2.0169 - mask_estimation_loss: 1.9946
Epoch 34/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -2.1246 - loss: -0.1214 - mask_estimation_loss: 2.0033
Epoch 35/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -4.7492 - loss: -2.7535 - mask_estimation_loss: 1.9957
Epoch 36/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -2.9504 - loss: -0.9536 - mask_estimation_loss: 1.9969
Epoch 37/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -2.2041 - loss: -0.2044 - mask_estimation_loss: 1.9997
Epoch 38/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -3.2450 - loss: -1.2449 - mask_estimation_loss: 2.0001
Epoch 39/50
```

```
2076/2076            6s 2ms/step -
feature_estimation_loss: -3.9249 - loss: -1.9267 - mask_estimation_loss: 1.9981
Epoch 40/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -5.8223 - loss: -3.8223 - mask_estimation_loss: 1.9999
Epoch 41/50
2076/2076            6s 2ms/step -
feature_estimation_loss: -7.3974 - loss: -5.3971 - mask_estimation_loss: 2.0004
Epoch 42/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -5.9039 - loss: -3.9016 - mask_estimation_loss: 2.0023
Epoch 43/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -11.0884 - loss: -9.0886 - mask_estimation_loss: 1.9998
Epoch 44/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -8.3581 - loss: -6.3571 - mask_estimation_loss: 2.0010
Epoch 45/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -6.9285 - loss: -4.9276 - mask_estimation_loss: 2.0011
Epoch 46/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -6.4271 - loss: -4.4238 - mask_estimation_loss: 2.0032
Epoch 47/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -6.2415 - loss: -4.2373 - mask_estimation_loss: 2.0042
Epoch 48/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -11.8180 - loss: -9.8223 - mask_estimation_loss: 1.9957
Epoch 49/50
2076/2076            4s 2ms/step -
feature_estimation_loss: -5.3685 - loss: -3.3696 - mask_estimation_loss: 1.9990
Epoch 50/50
2076/2076            5s 2ms/step -
feature_estimation_loss: -7.4109 - loss: -5.4054 - mask_estimation_loss: 2.0056
```

```
Model: "functional"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 35)	0	-
dense (Dense)	(None, 35)	1,260	input_layer[0][0]

```
mask_estimation (Dense)     (None, 35)                    1,260 ␣
↪dense[0][0]

feature_estimation          (None, 35)                    1,260 ␣
↪dense[0][0]
(Dense)                                                         ␣
↪
```

Total params: 7,562 (29.54 KB)

Trainable params: 3,780 (14.77 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 3,782 (14.78 KB)

[ ]:
```
encoder_path = "/content/encoder_model.keras"


encoder_model.save(encoder_path)
```

[ ]:
```
from keras.models import load_model
encoder_model = load_model(encoder_path)
```

[ ]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from keras.models import load_model

# ... (Load your data and define exclude_columns as before) ...
# Exclude specified columns
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual']
data_filtered = data.drop(columns=exclude_columns)

# Get the column names used during training
training_columns = data_filtered.columns  # Assuming data_filtered was used for␣
 ↪training

# Select the same columns from x_train and x_test
x_train_filtered = x_train[training_columns]
x_test_filtered = x_test[training_columns]
```

```
# Standardize using the same scaler used during training
# Assuming you saved the scaler, otherwise recreate it with the same parameters
# scaler = load_scaler("path/to/scaler.pkl")  # If saved
scaler = StandardScaler()  # If not saved, recreate it
x_train_scaled = scaler.fit_transform(x_train_filtered)
x_test_scaled = scaler.transform(x_test_filtered)

# Load the encoder model
encoder_model = load_model(encoder_path)

# Now predict using the correctly preprocessed data
X_train_scaled_encoded = encoder_model.predict(x_train_scaled)
X_test_scaled_encoded = encoder_model.predict(x_test_scaled)

# ... (Rest of your code) ...

logistic_model = LogisticRegression(max_iter=5000)
logistic_model.fit(X_train_scaled_encoded, y_train)
y_encoded = logistic_model.predict_proba(X_test_scaled_encoded)

from sklearn.metrics import log_loss
print("Logistic Regression Log Loss:", log_loss(y_test, y_encoded))

xgb_model = XGBClassifier(eval_metric='mlogloss')
xgb_model.fit(X_train_scaled_encoded, y_train)
y_encoded_xgb = xgb_model.predict_proba(X_test_scaled_encoded)

print("XGBoost Log Loss:", log_loss(y_test, y_encoded_xgb))
```

```
2279/2279             3s 1ms/step
977/977             1s 1ms/step
Logistic Regression Log Loss: 0.7689957107930324
XGBoost Log Loss: 0.8609980859973267
```

#Overview of Function

Function for the model, train, semi_supervised

[ ]:
```
import tensorflow as tf
from tensorflow.keras import layers, models, optimizers
import numpy as np  # Ensure numpy is imported

import tensorflow as tf
from tensorflow.keras import layers, models, optimizers

# Define the model function
```

```
def build_model(input_dimension, hidden_dimension, label_dimension,␣
 ↪activation=tf.nn.relu):
    inputs = tf.keras.Input(shape=(input_dimension,), name='model_input')
    x = layers.Dense(hidden_dimension, activation=activation,␣
 ↪name='model_dense_layer_1')(inputs)
    x = layers.Dense(hidden_dimension, activation=activation,␣
 ↪name='model_dense_layer_2')(x)
    y_logit = layers.Dense(label_dimension, activation=None,␣
 ↪name='model_logit_output')(x)
    y = layers.Activation('softmax', name='model_output')(y_logit)
    model = models.Model(inputs=inputs, outputs=[y_logit, y], name="model")
    return model

# Define the training function
def train(feature_batch, label_batch, unlabeled_feature_batch, model, beta,␣
 ↪supv_loss_fn, optimizer):
    with tf.GradientTape() as tape:
        y_logit, y = model(feature_batch, training=True)
        y_loss = supv_loss_fn(label_batch, y)

        unlabeled_y_logit, unlabeled_y = model(unlabeled_feature_batch,␣
 ↪training=True)
        unlabeled_y_loss = tf.reduce_mean(tf.nn.moments(unlabeled_y_logit,␣
 ↪axes=0)[1])

        total_loss = y_loss + beta * unlabeled_y_loss
        grads = tape.gradient(total_loss, model.trainable_weights)
    optimizer.apply_gradients(zip(grads, model.trainable_weights))
    return total_loss

# Define the semi-supervised function
def semi_supervised(x_train, y_train, x_unlabeled, x_test, parameters,␣
 ↪mask_probability, K, beta):
    hidden_dimension = parameters['hidden_dimension']
    batch_size = parameters['batch_size']
    epochs = parameters['epochs']
    input_dimension = x_train.shape[1]
    label_dimension = len(np.unique(y_train)) if y_train.ndim == 1 else y_train.
 ↪shape[1]

    # Map class labels if y_train is categorical
    if y_train.ndim == 1 or y_train.shape[1] == 1:
        class_mapping = {label: idx for idx, label in enumerate(np.
 ↪unique(y_train))}
        y_train = np.vectorize(class_mapping.get)(y_train)
```

```
    # Split training data into training and validation sets
    index = np.random.permutation(x_train.shape[0])
    train_index = index[:int(len(index) * 0.9)]
    valid_index = index[int(len(index) * 0.9):]

    splitted_train_x = x_train[train_index, :]
    splitted_train_y = y_train[train_index]
    splitted_valid_x = x_train[valid_index, :]
    splitted_valid_y = y_train[valid_index]

    # Data encoding
    encoder_model_path = "/content/encoder_model.keras"
    encoder = tf.keras.models.load_model(encoder_model_path)

    x_valid_encoded = encoder.predict(splitted_valid_x)
    x_test_encoded = encoder.predict(x_test)

    # Initialize the supervised learning model
    supervised_model = build_model(
        input_dimension=encoder.output_shape[1],
        hidden_dimension=hidden_dimension,
        label_dimension=label_dimension
    )
    optimizer = optimizers.Adam()
    supv_loss_fn = tf.keras.losses.CategoricalCrossentropy(from_logits=True)

    for epoch in range(epochs):
        batch_index = np.random.choice(len(splitted_train_x), batch_size,␣
↪replace=False)
        batch_x = splitted_train_x[batch_index]
        batch_y = splitted_train_y[batch_index]
        batch_x_encoded = encoder.predict(batch_x)

        batch_unlabeled_index = np.random.choice(len(x_unlabeled), batch_size,␣
↪replace=False)
        batch_unlabeled_x = x_unlabeled[batch_unlabeled_index]

        batch_unlabeled_x_shuffled = []
        for _ in range(K):
            mask_batch_unlabeled = binary_mask(mask_probability,␣
↪batch_unlabeled_x)
            _, unlabeled_shuffled_temp = corruption(mask_batch_unlabeled,␣
↪batch_unlabeled_x)
            unlabeled_shuffled_temp_encoded = encoder.
↪predict(unlabeled_shuffled_temp)
            batch_unlabeled_x_shuffled.append(unlabeled_shuffled_temp_encoded)
```

```
        batch_unlabeled_x_shuffled = np.concatenate(batch_unlabeled_x_shuffled,␣
 ↪axis=0)

        total_loss = train(batch_x_encoded, batch_y,␣
 ↪batch_unlabeled_x_shuffled, supervised_model, beta, supv_loss_fn, optimizer)

        y_valid_logit, y_valid = supervised_model(x_valid_encoded,␣
 ↪training=False)
        y_valid_loss = supv_loss_fn(splitted_valid_y, y_valid_logit)

        if epoch % 100 == 0:
            print(f"Epoch: {epoch}/{epochs}, Validation Loss: {y_valid_loss:.
 ↪4f}")

    y_test_logit, y_test = supervised_model(x_test_encoded, training=False)
    return y_test_logit, supervised_model
```

[ ]:
```
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split

df=data

# 1. Define features and labels
exclude_columns = ['Event', 'Time', 'Cell_length', 'file_number',␣
 ↪'event_number', 'label', 'individual'] # Replace 'target_label' with your␣
 ↪label column name
features = [col for col in df.columns if col not in exclude_columns]  # Exclude␣
 ↪specific columns

X = df[features].values  # Features (exclude the label column)
y = df['label'].values  # Labels (use the excluded column for labels)

# 2. Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# 3. Scale the features
scaler = StandardScaler()
new_df_scaled_train = scaler.fit_transform(X_train)  # Scaled training features
new_df_scaled_test = scaler.transform(X_test)        # Scaled test features

# 4. Handle unlabeled data
X_unlabeled = X_train.copy()  # Using training data as a proxy for unlabeled␣
 ↪data
x_unlabeled_scaled = scaler.transform(X_unlabeled)
```

```
# 5. Encode the labels (if needed)
encoder = OneHotEncoder(sparse_output=False)  # Use 'sparse_output' instead of␣
↪'sparse'
y_train_encoded = encoder.fit_transform(y_train.reshape(-1, 1))
y_test_encoded = encoder.transform(y_test.reshape(-1, 1))

new_df_labels_train = y_train_encoded
new_df_labels_test = y_test_encoded

# 6. Call the semi_supervised function
parameters = {
    'hidden_dimension': 100,
    'batch_size': 128,
    'epochs': 1000
}
mask_probability = 0.3
alpha = 2.0
K = 3
beta = 1.0

y_test, supervised_model = semi_supervised(
    x_train=new_df_scaled_train,
    y_train=new_df_labels_train,
    x_unlabeled=x_unlabeled_scaled,
    x_test=new_df_scaled_test,
    parameters=parameters,
    mask_probability=mask_probability,
    K=K,
    beta=beta
)
```

```
665/665             1s 2ms/step
1661/1661             4s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/backend/tensorflow/nn.py:593:
UserWarning: "`categorical_crossentropy` received `from_logits=True`, but the
`output` argument was produced by a Softmax activation and thus does not
represent logits. Was this intended?
  output, from_logits = _get_logits(
```

```
Epoch: 0/1000, Validation Loss: 6.2011
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4              0s 3ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 3ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
4/4              0s 2ms/step
```

```
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 4ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
Epoch: 100/1000, Validation Loss: 1.4652
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 3ms/step
4/4                0s 3ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
4/4                0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 200/1000, Validation Loss: 1.3058
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 7ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 6ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 4ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 300/1000, Validation Loss: 1.2685
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 7ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 400/1000, Validation Loss: 1.2485
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 6ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 7ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 500/1000, Validation Loss: 1.2529
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 600/1000, Validation Loss: 1.2069
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
Epoch: 700/1000, Validation Loss: 1.2375
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 6ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 6ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 6ms/step
4/4             0s 6ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
Epoch: 800/1000, Validation Loss: 1.1866
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
4/4               0s 2ms/step
4/4               0s 2ms/step
4/4               0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
Epoch: 900/1000, Validation Loss: 1.1833
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 6ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

```
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
```

```
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 2ms/step
4/4             0s 5ms/step
4/4             0s 5ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 3ms/step
4/4             0s 4ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
4/4             0s 2ms/step
```

[26]:
```python
import tensorflow as tf
from tensorflow.keras import layers, models

def model(input_dimension, hidden_dimension, label_dimension, activation=tf.nn.
 ↪relu):
    # inputs -> takes input dimension as argument
    inputs = tf.keras.Input(shape=input_dimension, name='model_input')
    x = layers.Dense(hidden_dimension, activation=activation,␣
 ↪name='model_dense_layer_1')(inputs) # dense layer 1
    x = layers.Dense(hidden_dimension, activation=activation,␣
 ↪name='model_dense_layer_2')(x) # dense layer 2
    y_logit = layers.Dense(label_dimension, activation=None,␣
 ↪name='model_logit_output')(x) # logit output
    y = layers.Activation('softmax', name='model_output')(y_logit) # actual␣
 ↪prediction
    model = models.Model(inputs=inputs, outputs=[y_logit, y], name="model") #␣
 ↪model creation
    return model
def train(feature_batch, label_batch, unlabeled_feature_batch, model, beta,␣
 ↪supv_loss_fn, optimizer):
    with tf.GradientTape() as tape:
        y_logit, y = model(feature_batch, training=True) # getting outputs for␣
 ↪labeled data
        y_loss = supv_loss_fn(label_batch, y) # calculating supervised loss␣
 ↪function for labeled data
        unlabeled_y_logit, unlabeled_y = model(unlabeled_feature_batch,␣
 ↪training=True) # getting outputs for unlabeled data
```

```
        unlabeled_y_loss = tf.reduce.mean(tf.nn.moments(unlabeled_y_logit,␣
↪axes=0)[1]) # loss function for unlabeled data
        # unsupervised loss function calculates the mean and variance of the␣
↪outputs and will penalize if the variance is high i.e, it will try to reduce␣
↪the variance of the output
        total_loss = y_loss + beta * unlabeled_y_loss # loss formula. Beta is a␣
↪hyperparameter i.e, you have to enter your own value for this
        grads = tape.gradient(total_loss, model.trainable_weights) #␣
↪calculating gradients or by how much the weights need to be changed
    optimizer.apply_gradients(zip(grads, model.trainable_weights)) # making the␣
↪changes to the weights

    return total_loss

def semi_supervised(x_train, y_train, x_unlabeled, x_test, parameters,␣
↪mask_probability, K, beta):
    # parameters is a dictionary
    hidden_dimension = parameters['hidden_dimension']
    act_fn = tf.nn.relu
    batch_size = parameters['batch_size']
    epochs = parameters['epochs']
    input_dimension = x_train.shape[1]
    label_dimension = y_train.shape[1]
    total_classes = len(np.unique(y_train)) #calculates the total number of␣
↪classes

    classes_dimension = total_classes


    if y_train.ndim == 1 or y_train.shape[1] == 1:

      total_classes = len(np.unique(y_train)) # calculates the total number of␣
↪classes

      classes_dimension = total_classes

      class_mapping = {label: idx for idx, label in enumerate(np.
↪unique(y_train))} # maps each unique class label in y_train to an integer␣
↪index

      y_train_mapped = np.vectorize(class_mapping.get)(y_train) # using np.
↪vectorize to apply class_mapping function to each element in y_train. This␣
↪maps each original label to its integer index
      # for ex in our case, if the classes are from 1-15 ---> they will be␣
↪mapped to 0 - 14 where 1 -> 0 , 2 -> 1 , 3 -> 2 and so on
```

```
  else:
    print("y_train is not of dimension. Please check your input argument for y_train")
    class_mapping = None
    classes_dimension = y_train.shape[1]
    y_train_mapped = y_train


  # Splitting the training data into training and validation sets
  index = np.random.permutation(x_train.shape[0])
  train_index = index[:int(len(index) * 0.9)]
  valid_index = index[int(len(index) * 0.9):]

  # Getting training sets
  splitted_train_x = x_train[train_index, :]
  splitted_train_y = y_train_mapped[train_index]

  # Getting validation sets
  splitted_valid_x = x_train[valid_index, :]
  splitted_valid_y = y_train_mapped[valid_index]

  # Data encoding
  encoder = "/content/encoder_model.keras"

  x_valid_encoded = encoder.predict(splitted_valid_x)
  x_test_encoded = encoder.predict(x_test)

  # Initialize the model
  Model = model(input_dimension=(encoder.output_shape[1],),
                hidden_dimension=hidden_dimension,
                label_dimension=label_dimension)
  optimizer = optimizers.Adam()
  supv_loss_fn = tf.keras.losses.CategoricalCrossentropy(from_logits=True)

  # Training loop
  for epoch in range(total_epochs):
    # Sending a batch of labeled data
    batch_index = np.random.choice(len(x_train), batch_size, replace=False)
    batch_x = x_train[batch_index]
    batch_y = y_train[batch_index]
    batch_x_encoded = encoder.predict(batch_x)

    # Sending a batch on unlabeled data
    batch_unlabeled_index = np.random.choice(len(x_unlabeled), batch_size, replace=False)
    batch_unlabeled_x = x_unlabeled[batch_unlabeled_index]
    batch_unlabeled_x_shuffled = []
```

```
    for rep in range(K):

      mask_batch_unlabeled = mask_generator(mask_probability,␣
↪batch_unlabeled_x)  # Generate mask for batch of unlabeled data

      _, unlabeled_shuffled_temp = corruption(mask_batch_unlabeled,␣
↪batch_unlabeled_x)  # Get shuffled batch data

      unlabeled_shuffled_temp_encoded = encoder.
↪predict(unlabeled_shuffled_temp)  # Encode the shuffled data

      batch_unlabeled_x_shuffled.append(unlabeled_shuffled_temp_encoded)  #␣
↪Append the encoded data to the original unlabeled shuffled array

  batch_unlabeled_x_shuffled = np.concatenate(batch_unlabeled_x_shuffled,␣
↪axis=0)  # This line is outside the for loop
  # Make sure you add all the values to shuffled data
  total_loss = train(batch_x_encoded, batch_y, batch_unlabeled_x_shuffled,␣
↪Model, beta, supv_loss_fn, optimizer)

  y_valid_logit, y_valid = Model(x_valid_encoded, training=False)
  y_valid_loss = supv_loss_fn(splitted_valid_y, y_valid_logit)  # Compute␣
↪loss for supervised loss function

  if epoch % 100 == 0:  # Print loss after every 100 epochs
      print(f"epoch: {epoch}/{epochs}, Validation Loss: {y_valid_loss:.4f}")

      # Get the outputs for testing data
  y_test_logit, y_test = Model(x_test_encoded, training=False)

  return y_test_logit, Model
```

[28]:
```
## Perf Metric
```

[30]:
```
from sklearn.metrics import accuracy_score, roc_auc_score
from sklearn.preprocessing import label_binarize
import numpy as np

def perf_metric(metric_type, y_true, y_pred):
    """
    Evaluate the performance of a classification model.

    Parameters:
        metric_type (str): The metric to calculate ('acc' for accuracy, 'auc'␣
↪for AUROC).
```

```
        y_true (ndarray): True labels (one-hot encoded or class indices).
        y_pred (ndarray): Predicted probabilities (one-hot encoded or␣
↪probabilities).

    Returns:
        float: Calculated performance metric.
    """
    # Decode one-hot encoded labels if needed
    if y_true.ndim > 1:  # If one-hot encoded, convert to class indices
        y_true = np.argmax(y_true, axis=1)
    if y_pred.ndim > 1:  # If one-hot encoded, convert to class indices for␣
↪accuracy
        y_pred_classes = np.argmax(y_pred, axis=1)

    if metric_type == 'acc':
        # Accuracy Calculation
        return accuracy_score(y_true, y_pred_classes)

    elif metric_type == 'auc':
        # AUROC Calculation
        # Binarize true labels for AUROC calculation if needed
        y_true_binarized = label_binarize(y_true, classes=np.unique(y_true))

        if y_pred.shape[1] == 2:  # Binary classification
            y_pred_binary = y_pred[:, 1]  # Probability of the positive class
            return roc_auc_score(y_true_binarized[:, 1], y_pred_binary)
        else:  # Multiclass classification
            return roc_auc_score(y_true_binarized, y_pred, average='macro',␣
↪multi_class='ovr')

    else:
        raise ValueError("Unsupported metric_type. Use 'acc' for accuracy or␣
↪'auc' for AUROC.")


# Decode one-hot encoded predictions and true labels for example
y_true = new_df_labels_test  # True labels
y_pred = y_test  # Model predicted probabilities

# Calculate Accuracy
accuracy = perf_metric('acc', y_true, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# Calculate AUROC
auroc = perf_metric('auc', y_true, y_pred)
print(f"AUROC: {auroc:.4f}")
```

```
Accuracy: 0.5306
AUROC: 0.7517
```

[ ]: