Analysis of Signal and Background Data

In this section, we load data from two Excel files, representing signal and background datasets. The data is read using the `pandas` library, which provides powerful data analysis and manipulation tools. Below is the code snippet used for loading the data, as well as a preview of the first few rows from each dataset.

In [1]:
```
import pandas as pd

# Load data from the Excel files
signal_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/SignalTTr
background_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/HBack

# Display the first few rows of each DataFrame to understand their struct
print(signal_df.head())
print(background_df.head())
```

```
     jet1_pt     jet2_pt     jet3_pt     jet4_pt     jet5_pt     jet6_pt  
\
0  173.034775  156.897537   86.575188   71.896362   68.743156   64.994034   
1  140.163620  134.187286  116.484482   92.105598   66.637825   64.564697   
2  455.637024  316.880127  186.414948  184.964386  142.811707  135.840546   
3  284.727386  240.010071  232.987961  109.618019   97.330498   84.789009   
4  307.271576  250.632828  182.247269  116.416428   75.660553   71.928635   

   numJets  numBJets    bjet1_pt  sumbjet_pt  ...  missing_ET  rat_MET_HT  
\
0       14         3  156.897537  308.106750  ...  144.099930    4.705777   
1       10         4  134.187286  351.839783  ...   15.960485    0.569030   
2       14         5  316.880127  575.655579  ...   46.435917    1.053144   
3       15         2  240.010071  324.799072  ...   48.165051    1.246486   
4       11         4  250.632828  568.604431  ...    1.918720    0.055819   

   deltaEta_jets  deltaPhi_jets  deltaEta_bjets  deltaPhi_bjets  deltaR_je
ts  \
0       0.610216       0.767123       -0.797419        0.721574     0.9802
25   
1       0.595912       3.703912        0.160582       -3.763749     3.7515
43   
2       0.852900      -2.355428       -1.971783        3.390985     2.5050
90   
3       1.459665       3.615045       -2.135111       -0.397436     3.8986
12   
4       1.454572       2.976023       -0.322397       -0.520300     3.3124
75   

   deltaR_bjets    top_mass        Weight  
0      1.075428  165.831696  4.483259e-07  
1      3.767173  170.195282  4.483259e-07  
2      3.922589  176.549606  4.483259e-07  
3      2.171786  172.850143  4.483259e-07  
4      0.612088  164.243088  4.483259e-07  

[5 rows x 21 columns]
     jet1_pt     jet2_pt     jet3_pt    jet4_pt    jet5_pt    jet6_pt  \
0  111.475479   99.509056   83.813049  68.029373  44.013199  35.163448   
1  252.901352  218.618286   79.311409  52.913513  35.482716  35.259525   
2  227.598694  194.158051  122.720116  87.814682  81.503639  78.380226   
3   75.846504   55.767906   55.049553  52.443012  51.964230  48.103096   
4  161.462692   99.148483   88.145340  69.618416  26.885643  22.892559   

    bjet1_pt  sumbjet_pt   Scalar_HT  missing_ET  ...  numJets  numBJets  
\
0  111.475479  239.441330  522.894531   20.140335  ...        9         4   
1  252.901352  455.868530  736.121643   62.563381  ...        8         5   
2  227.598694  625.980469  881.370178   93.425819  ...        8         4   
3   75.846504  127.810730  417.931366   12.673064  ...        8         2   
4   99.148483  168.766907  468.153137   31.030519  ...        6         2   

   deltaEta_jets  deltaPhi_jets  deltaEta_bjets  deltaPhi_bjets  deltaR_je
ts  \
0      -0.598871      -2.744731       -0.302492       -3.355825     2.8093
05   
1      -1.035444       2.813916       -1.061326        3.847212     2.9983
78   
2      -0.830490      -3.286546       -0.830490       -3.286546     3.3898
53   
```

```
3      -0.155088      1.098900      -1.406893       3.676342     1.1097
90
4      -0.820102      1.048206      -0.084306      -3.092073     1.3309
03

   deltaR_bjets    top_mass    Weight
0      3.369431  173.138046  0.156094
1      3.990921  170.884048  0.156094
2      3.389853  185.588913  0.156094
3      3.936348  175.873367  0.156094
4      3.093222  181.832993  0.156094

[5 rows x 21 columns]
```

Identifying Common Variables

In this section, we display the column names from both the signal and background datasets. This helps to identify common variables that may be used for further analysis, such as feature comparison or classification tasks. Understanding the structure of each dataset is crucial for the next steps in our analysis.

In [2]:

```python
# Display column names to identify common variables
print(signal_df.columns)
print(background_df.columns)
```

```
Index(['jet1_pt', 'jet2_pt', 'jet3_pt', 'jet4_pt', 'jet5_pt', 'jet6_pt',
       'numJets', 'numBJets', 'bjet1_pt', 'sumbjet_pt', 'Scalar_HT',
       'missing_ET', 'rat_MET_HT', 'deltaEta_jets', 'deltaPhi_jets',
       'deltaEta_bjets', 'deltaPhi_bjets', 'deltaR_jets', 'deltaR_bjets',
       'top_mass', 'Weight'],
      dtype='object')
Index(['jet1_pt', 'jet2_pt', 'jet3_pt', 'jet4_pt', 'jet5_pt', 'jet6_pt',
       'bjet1_pt', 'sumbjet_pt', 'Scalar_HT', 'missing_ET', 'rat_MET_HT',
       'numJets', 'numBJets', 'deltaEta_jets', 'deltaPhi_jets',
       'deltaEta_bjets', 'deltaPhi_bjets', 'deltaR_jets', 'deltaR_bjets',
       'top_mass', 'Weight'],
      dtype='object')
```

Visualizing Distributions of Key Variables in Signal and Background Data

Histograms

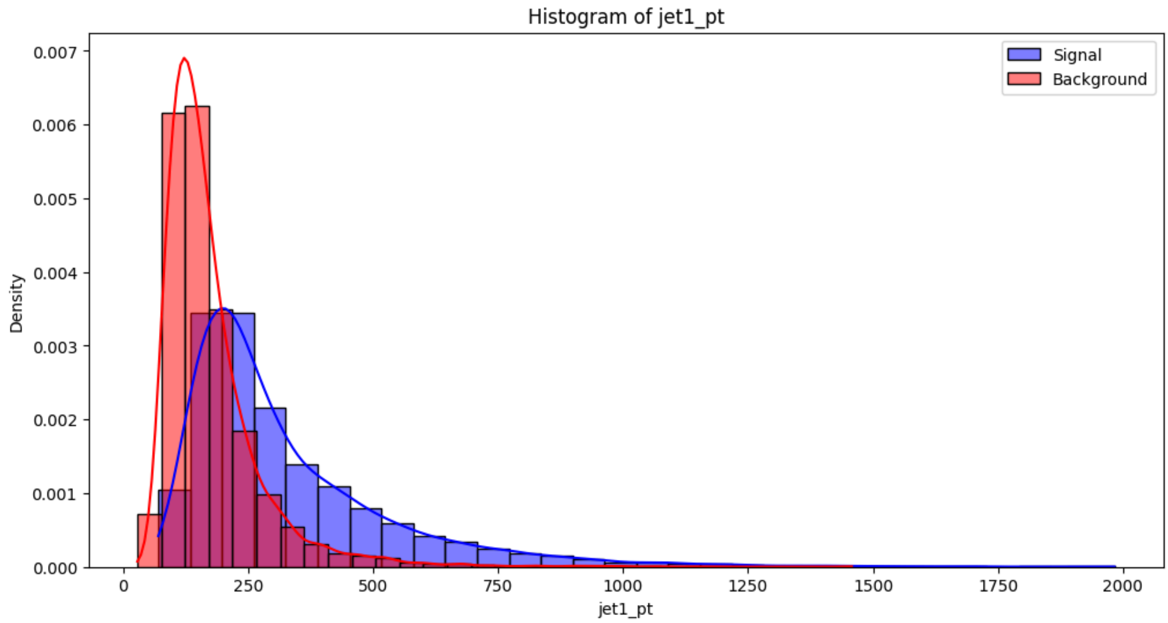
In this section, we plot histograms of selected variables to compare their distributions between the signal and background datasets. This helps to visually inspect which variables might be effective for distinguishing between the two classes. Each plot represents the distribution of a specific variable for both the signal (blue) and background (red) samples. The histograms are arranged in a 5x5 grid for a comprehensive overview.

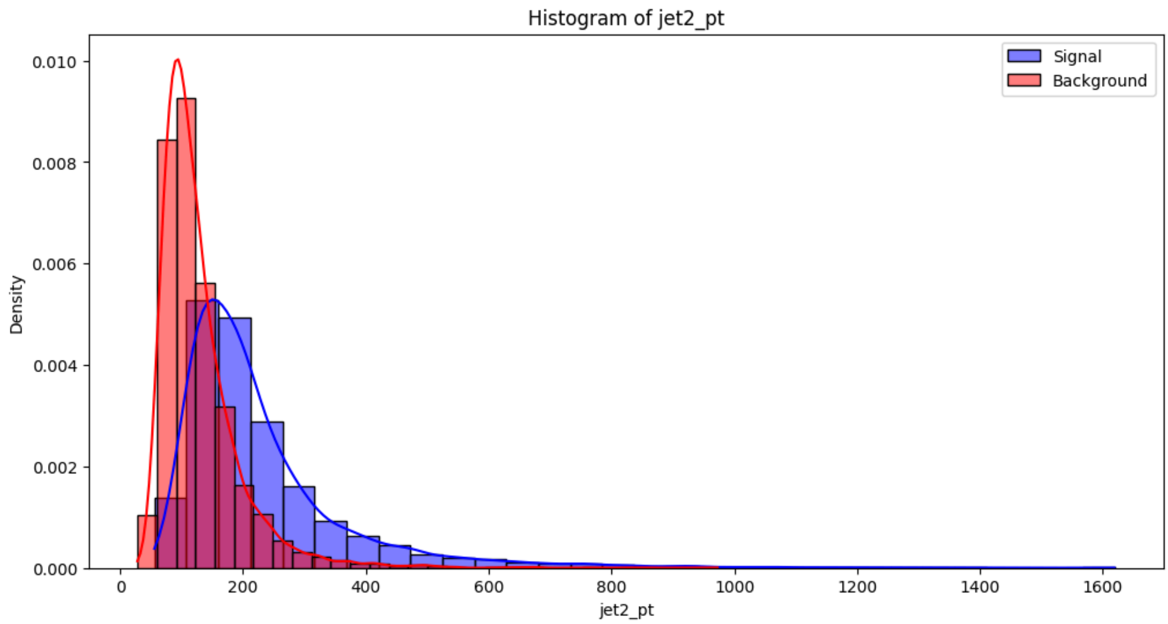
In [1]:

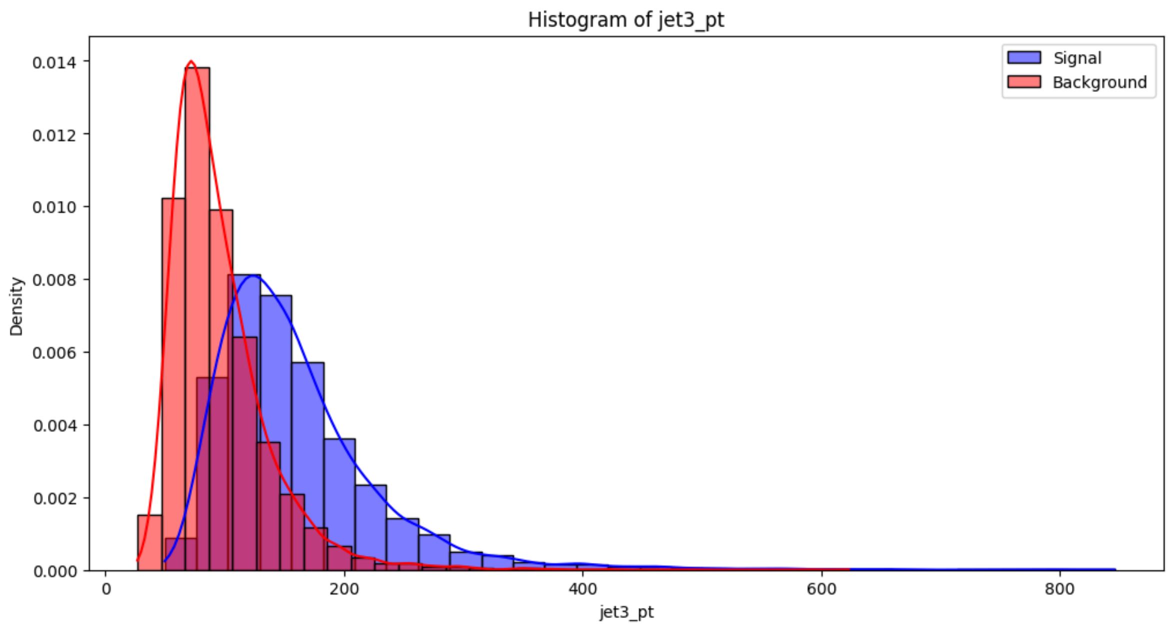
```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

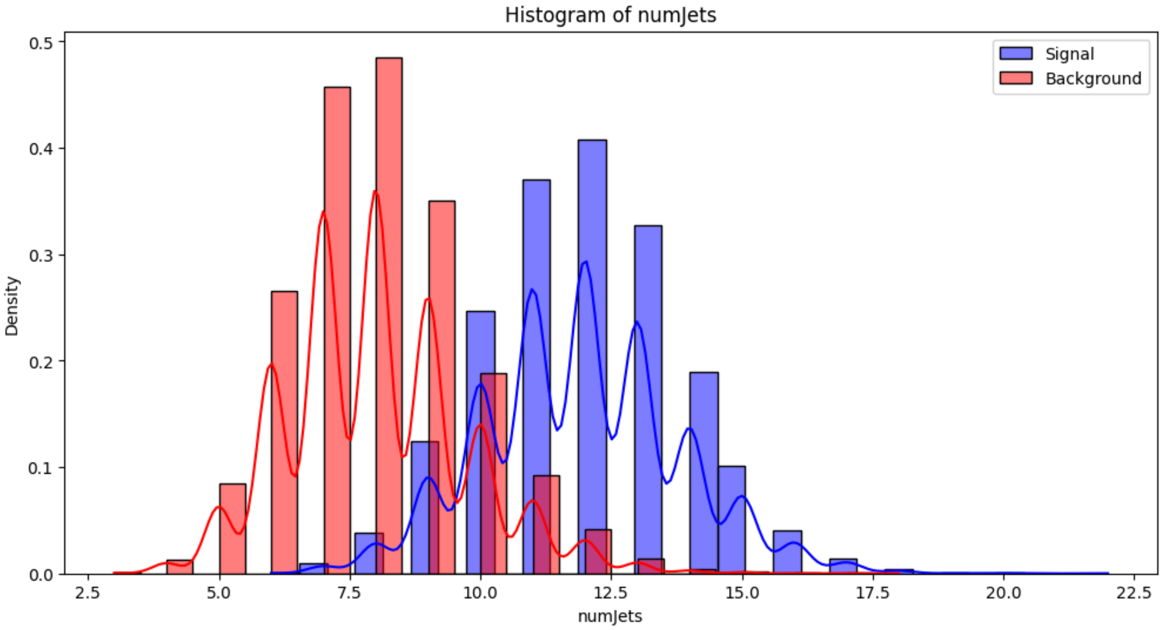
```
# Load your datasets
signal_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/SignalTTr
background_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/HBack


# Define the variables to analyze
variables = ['jet1_pt', 'jet2_pt', 'jet3_pt', 'numJets', 'numBJets',
             'bjet1_pt', 'sumbjet_pt', 'Scalar_HT', 'missing_ET',
             'rat_MET_HT', 'deltaEta_jets', 'deltaPhi_jets',
             'deltaEta_bjets', 'deltaPhi_bjets', 'deltaR_jets',
             'deltaR_bjets', 'top_mass']

# Initialize a list to collect statistics
summary_list = []

# Calculate statistics and create visualizations
for var in variables:
    signal_mean = signal_df[var].mean()
    background_mean = background_df[var].mean()
    signal_std = signal_df[var].std()
    background_std = background_df[var].std()

    # Append statistics to the summary list
    summary_list.append({
        "Variable": var,
        "Mean (Signal)": signal_mean,
        "Mean (Background)": background_mean,
        "Std Dev (Signal)": signal_std,
        "Std Dev (Background)": background_std,
        "Observations": ""
    })

    # Visualizations
    plt.figure(figsize=(12, 6))
    sns.histplot(signal_df[var], bins=30, color='blue', label='Signal', k
    sns.histplot(background_df[var], bins=30, color='red', label='Backgro
    plt.title(f'Histogram of {var}')
    plt.xlabel(var)
    plt.ylabel('Density')
    plt.legend()
    plt.savefig(f"{var}_histogram.png")  # Save histogram
    plt.show()

# Create a summary DataFrame from the list
summary = pd.DataFrame(summary_list)

# Print the summary DataFrame
print(summary)

# Save the summary to a text file
with open("analysis_summary.txt", "w") as f:
    f.write("# Data Analysis Results\n\n")
    f.write("## Variables Analyzed\n")
    f.write(", ".join(variables) + "\n\n")
    f.write("## Key Statistics\n\n")
    f.write(summary.to_markdown(index=False))  # Save summary in markdown
    f.write("\n\n## Insights from Visualizations\n")
    f.write("Refer to the saved histogram images for detailed insights.\n
```

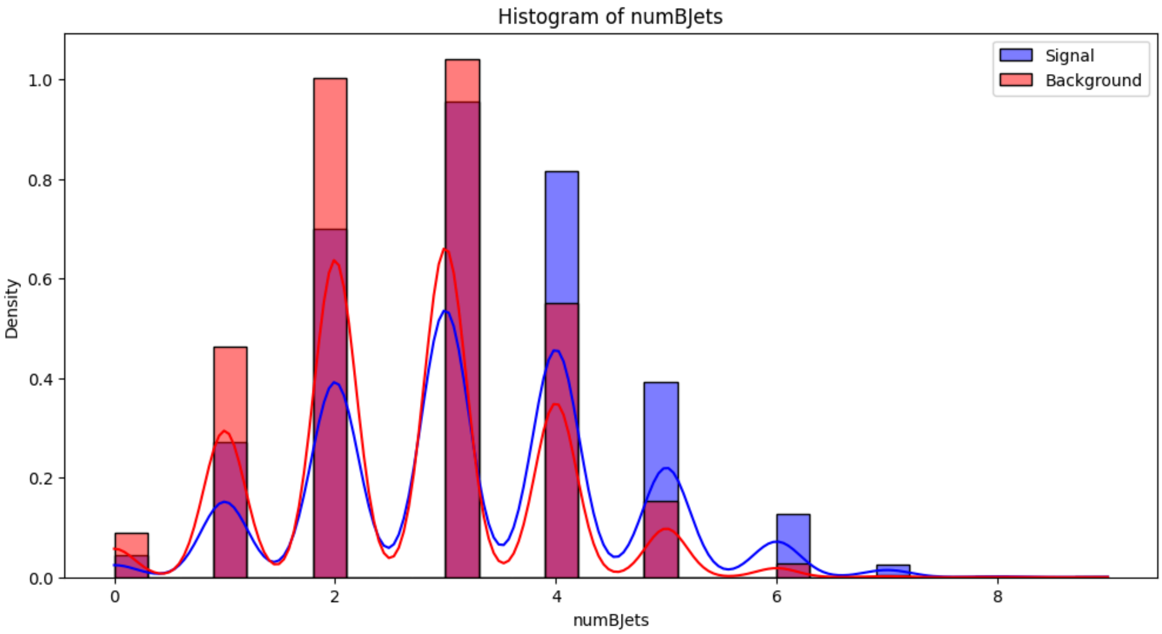
```
    f.write("\n\n## Conclusion\n")
    f.write("Please summarize your conclusions based on the analysis and
```

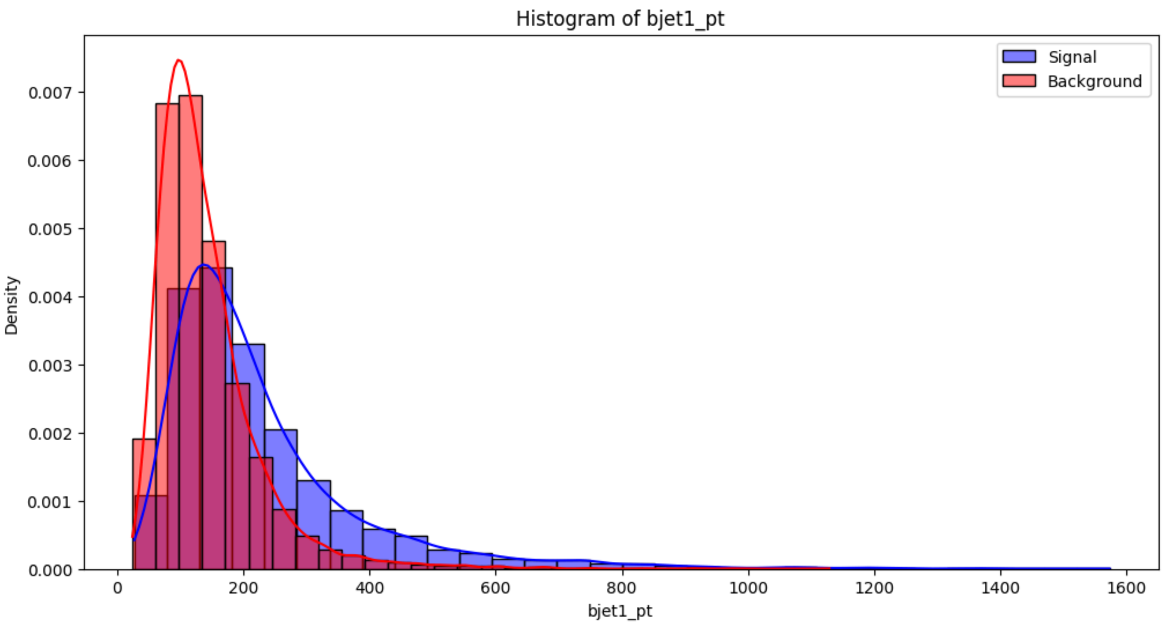


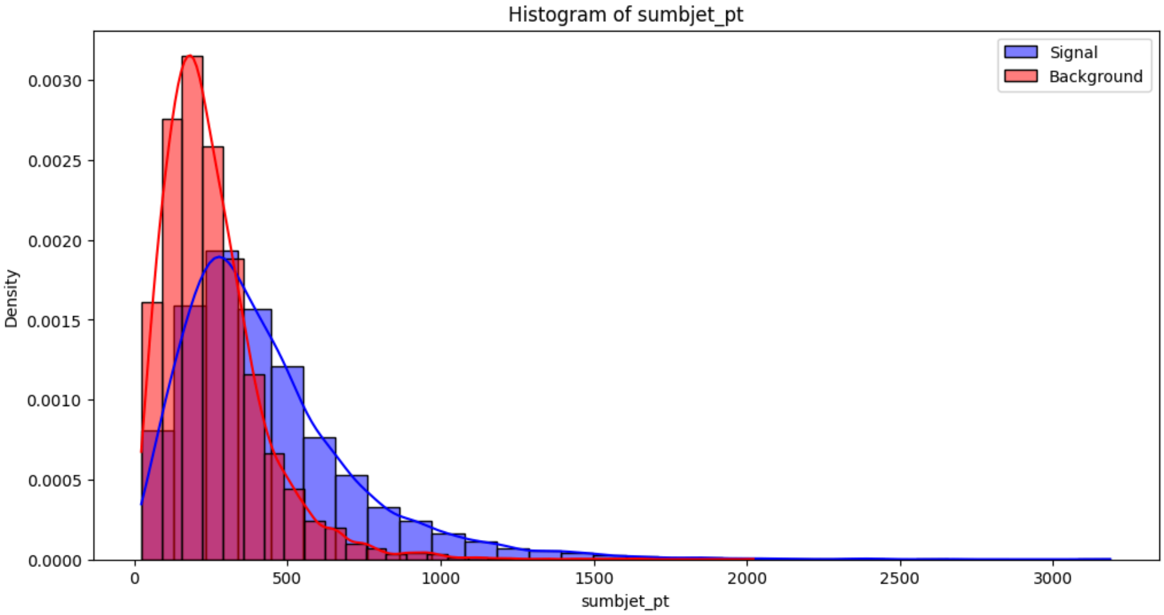


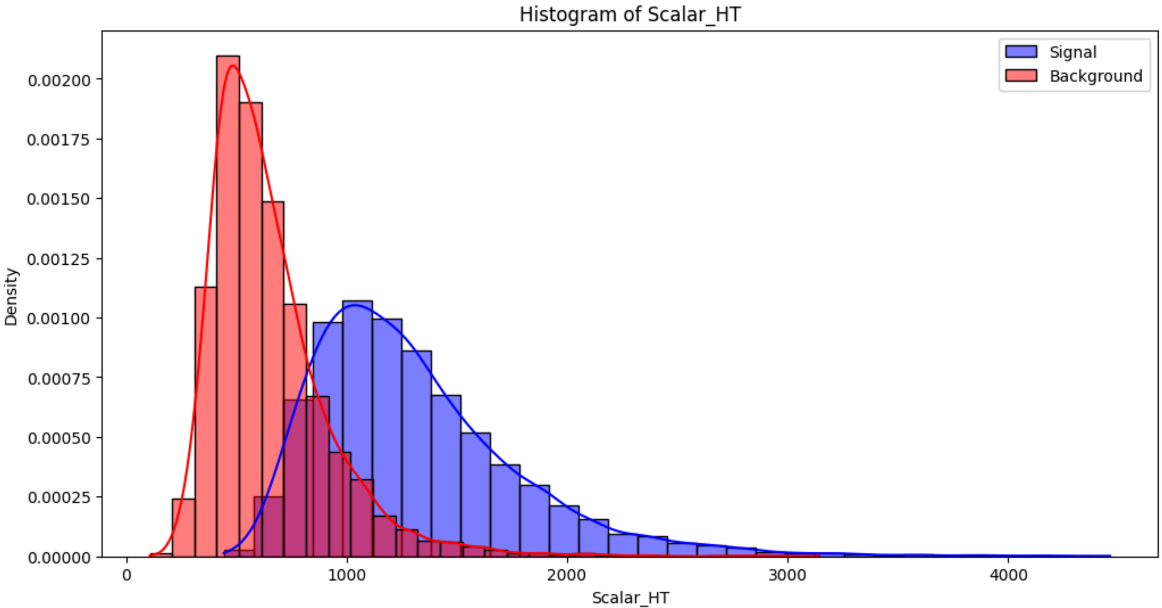


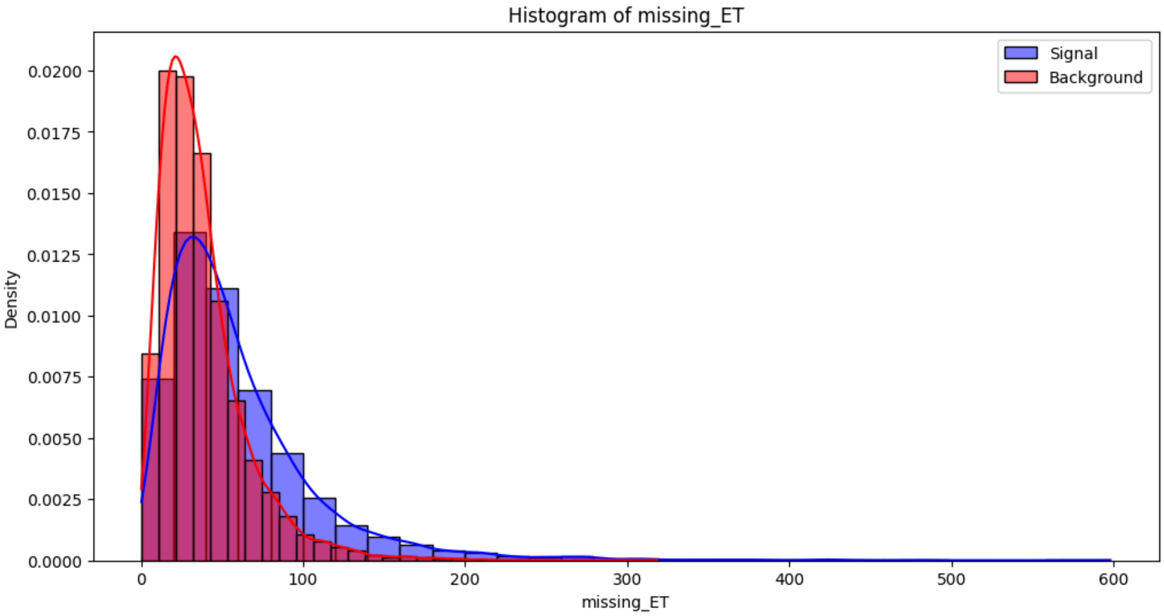

Histogram of numJets
Signal
Background
Density
numJets
0.0
0.1
0.2
0.3
0.4
0.5
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5

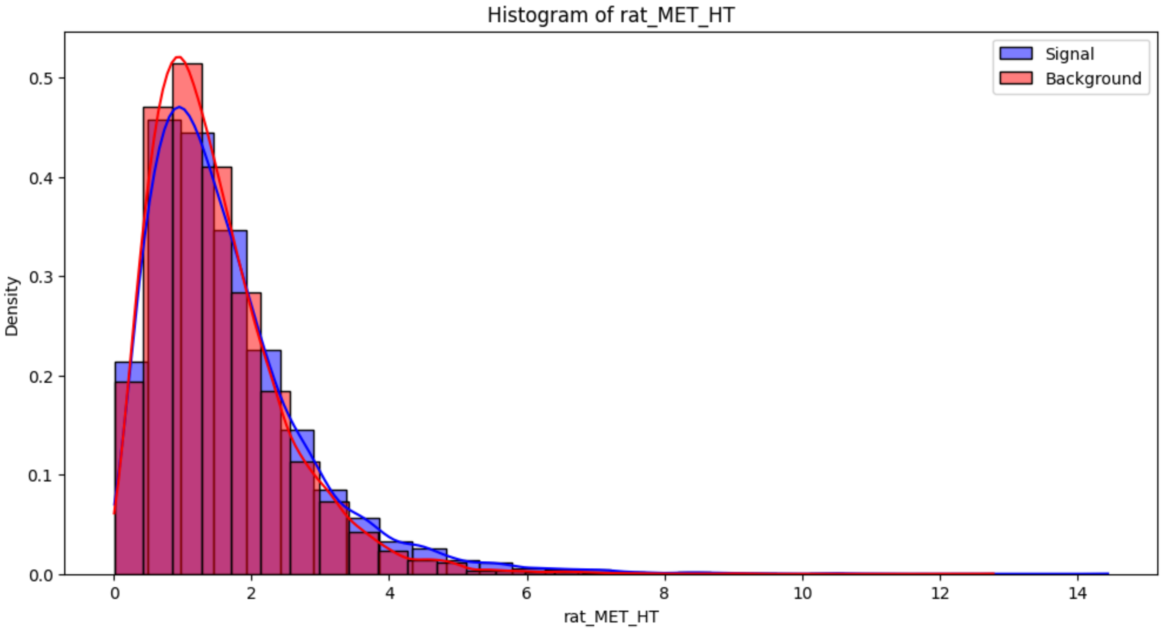

Histogram of numBJets
Signal
Background
Density
numBJets
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8

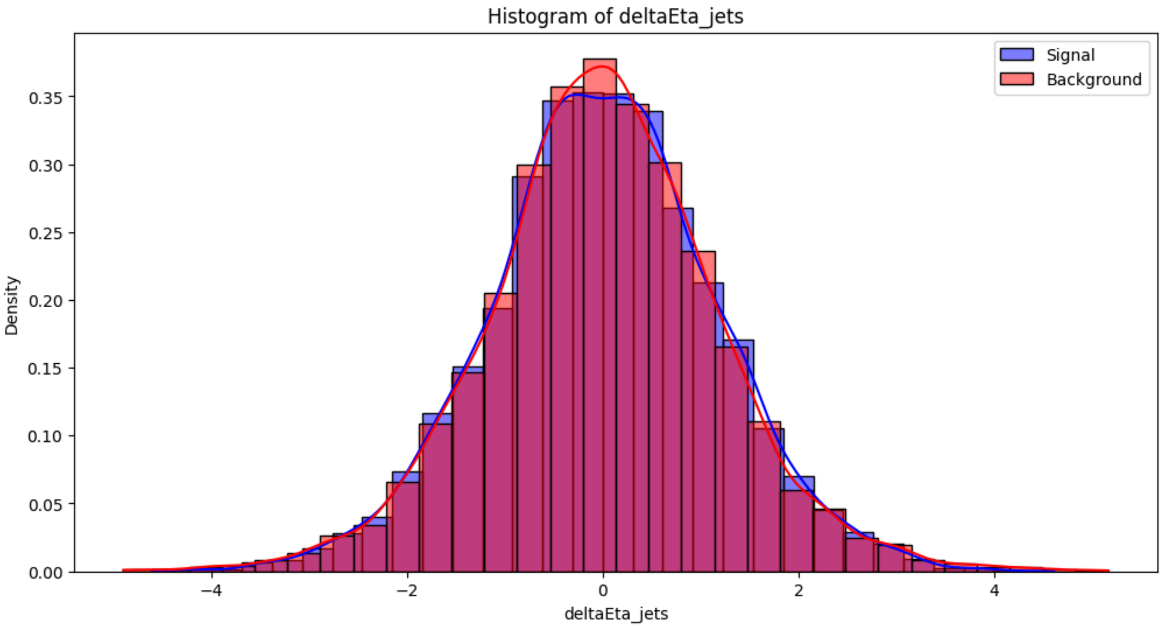

Histogram of bjet1_pt
Signal
Background
Density
bjet1_pt
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0
200
400
600
800
1000
1200
1400
1600

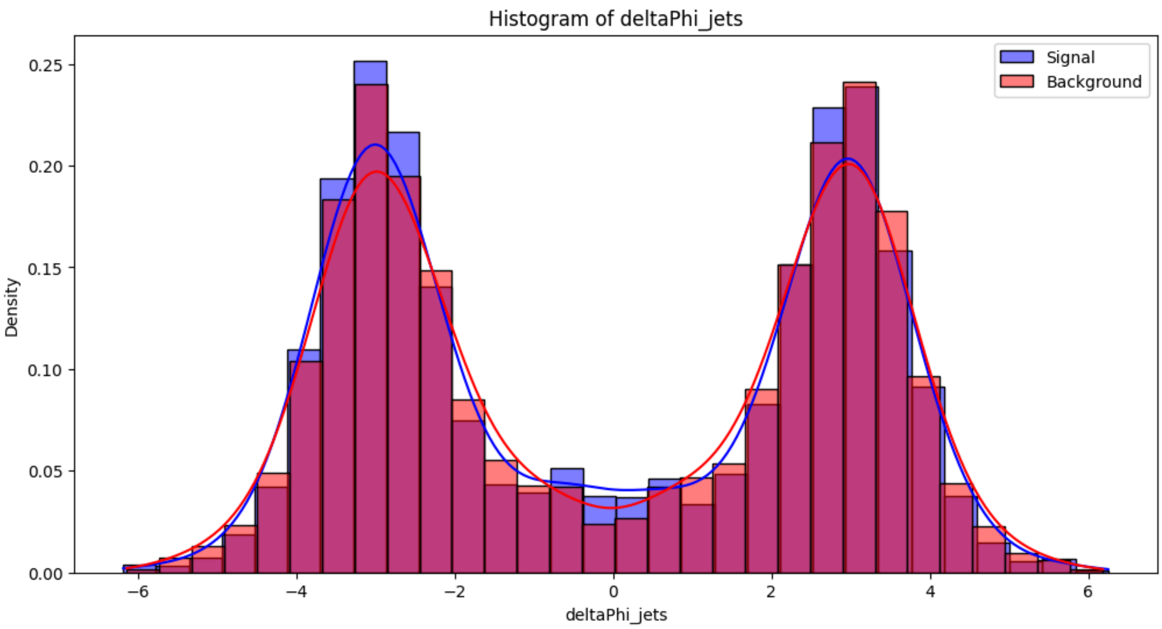

Histogram of sumbjet_pt
Signal
Background
Density
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
0.0030
0
500
1000
1500
2000
2500
3000
sumbjet_pt

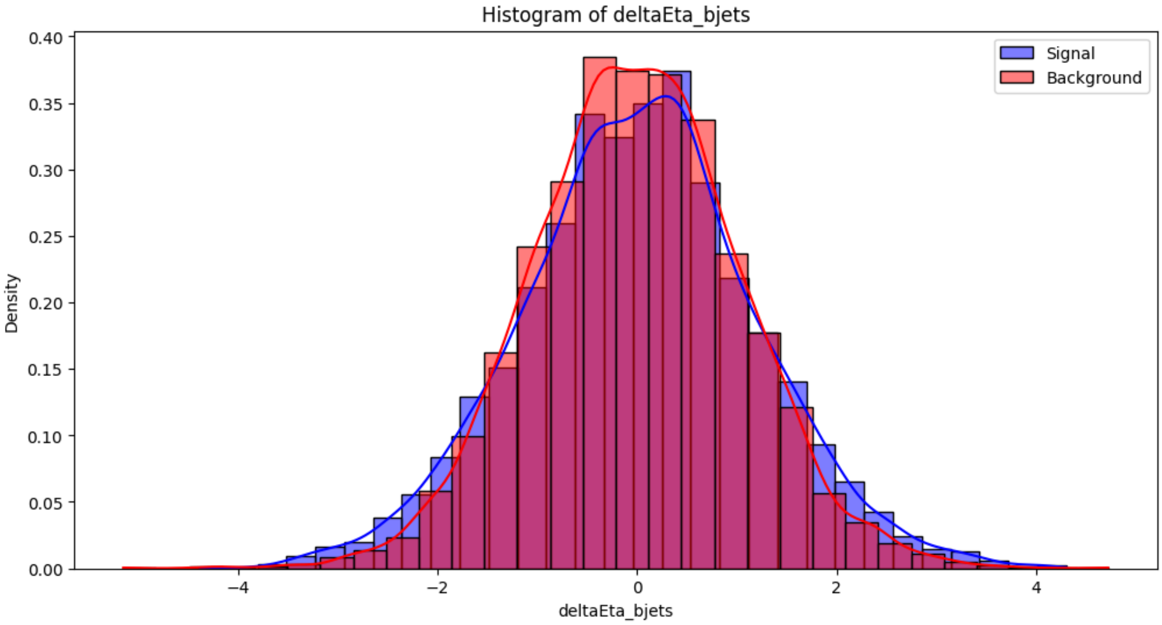

Histogram of Scalar_HT
Signal
Background
Density
0.00000
0.00025
0.00050
0.00075
0.00100
0.00125
0.00150
0.00175
0.00200
0
1000
2000
3000
4000
Scalar_HT

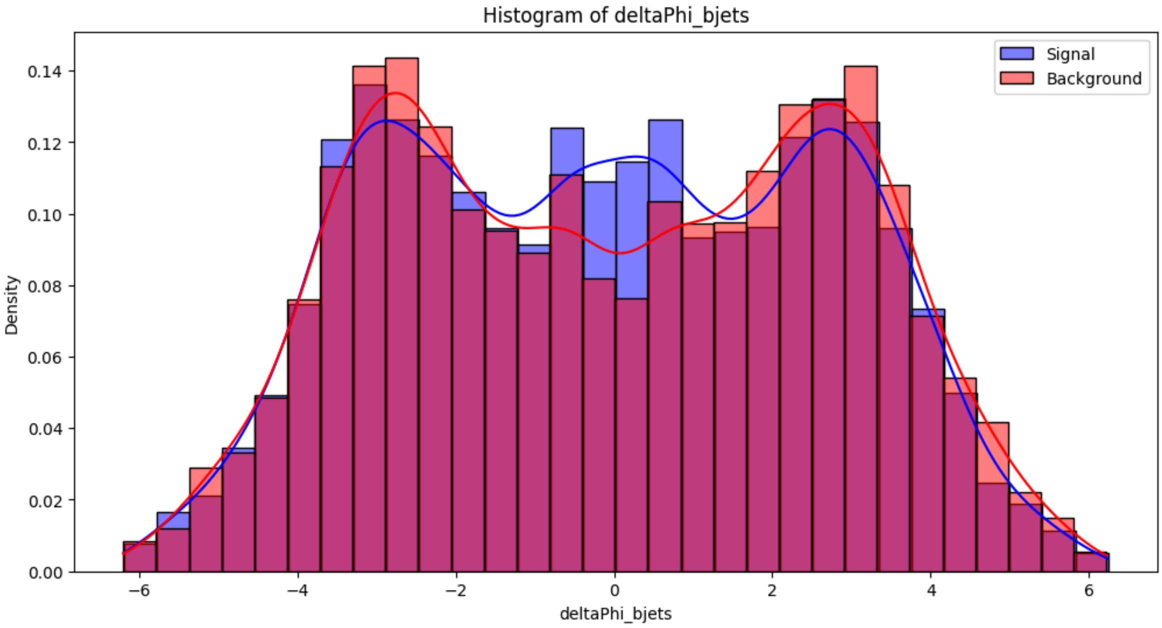

Histogram of missing_ET
Signal
Background
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
0
100
200
300
400
500
600
missing_ET

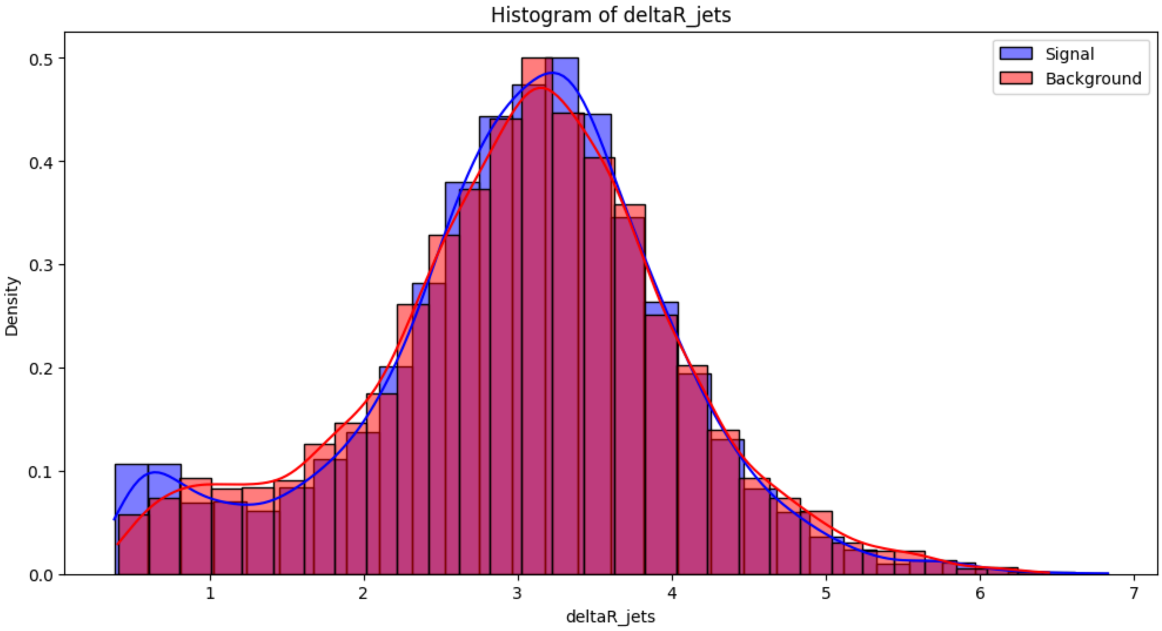

Histogram of rat_MET_HT
Signal
Background
Density
0.0
0.1
0.2
0.3
0.4
0.5
0
2
4
6
8
10
12
14
rat_MET_HT

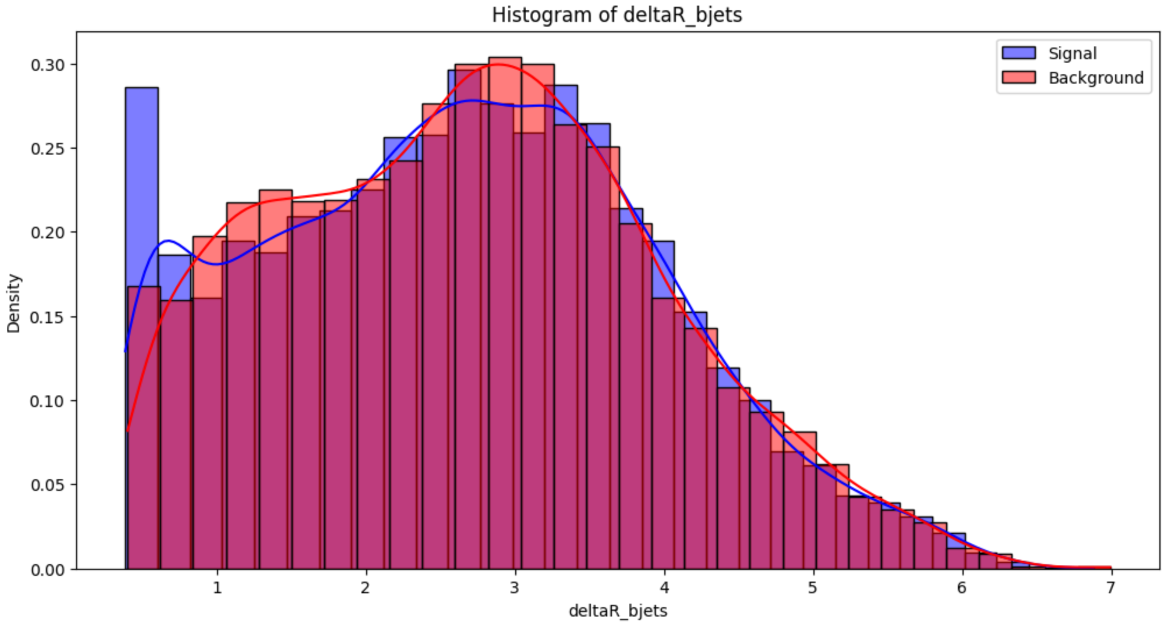

Histogram of deltaEta_jets
Signal
Background
Density
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
−4
−2
0
2
4
deltaEta_jets

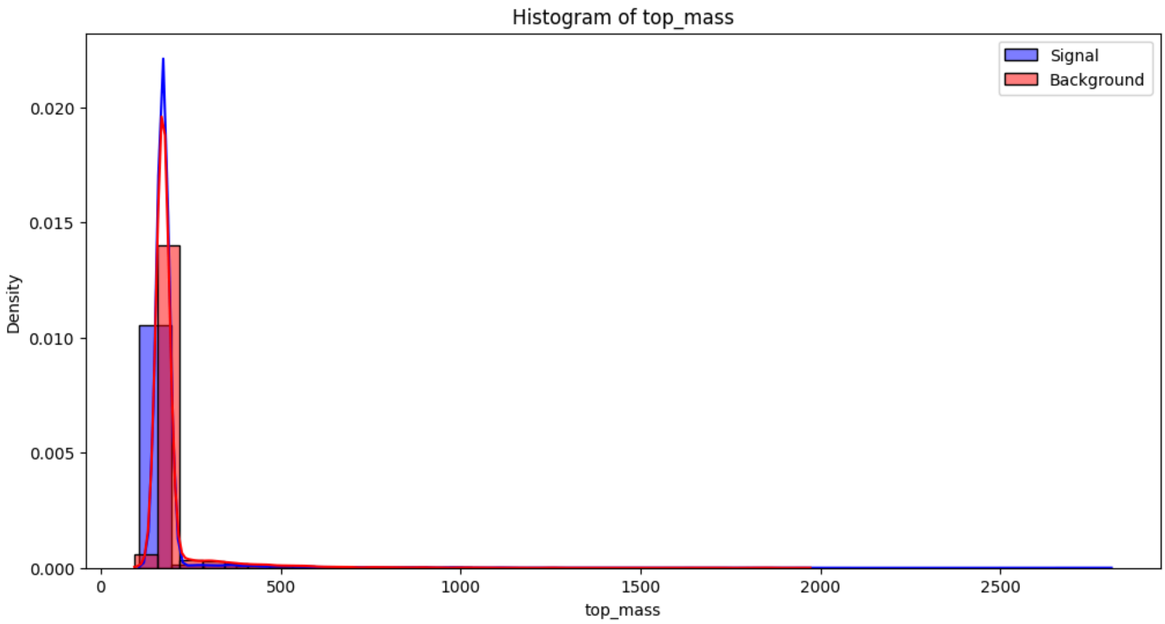

Histogram of deltaPhi_jets
Signal
Background
Density
0.00
0.05
0.10
0.15
0.20
0.25
−6
−4
−2
0
2
4
6
deltaPhi_jets


Histogram of deltaEta_bjets
Signal
Background
Density
deltaEta_bjets


Histogram of deltaPhi_bjets
Signal
Background
Density
deltaPhi_bjets


Histogram of deltaR_jets
Signal
Background
Density
deltaR_jets


Histogram of deltaR_bjets
Signal
Background
Density
0.00
0.05
0.10
0.15
0.20
0.25
0.30
1
2
3
4
5
6
7
deltaR_bjets


Histogram of top_mass
Signal
Background
Density
0.000
0.005
0.010
0.015
0.020
0
500
1000
1500
2000
2500
top_mass

	Variable	Mean (Signal)	Mean (Background)	Std Dev (Signal)	Std Dev (Background)	Observations
0	jet1_pt	327.913806	174.845906	210.909642	100.716293	
1	jet2_pt	227.726548	128.249269	138.746388	66.860378	
2	jet3_pt	160.075479	95.247804	70.473132	41.381533	
3	numJets	11.907100	7.992100	1.867784	1.695501	
4	numBJets	3.220000	2.628500	1.346618	1.188036	
5	bjet1_pt	228.028349	141.447562	160.211425	82.622212	
6	sumbjet_pt	431.375690	255.092182	293.190688	165.144226	
7	Scalar_HT	1307.858939	648.850664	471.347441	269.878435	
8	missing_ET	59.393203	37.499789	48.249683	27.502012	
9	rat_MET_HT	1.635210	1.487517	1.181753	0.981800	
10	deltaEta_jets	0.001854	-0.001165	1.149845	1.178140	
11	deltaPhi_jets	-0.052621	0.007301	2.938624	2.952025	
12	deltaEta_bjets	0.014526	-0.008204	1.178438	1.045917	
13	deltaPhi_bjets	-0.065385	0.009292	2.699535	2.786250	
14	deltaR_jets	2.990983	3.015709	1.006753	1.003550	
15	deltaR_bjets	2.641749	2.691509	1.304285	1.269779	
16	top_mass	188.448369	195.010709	99.765615	101.202783	

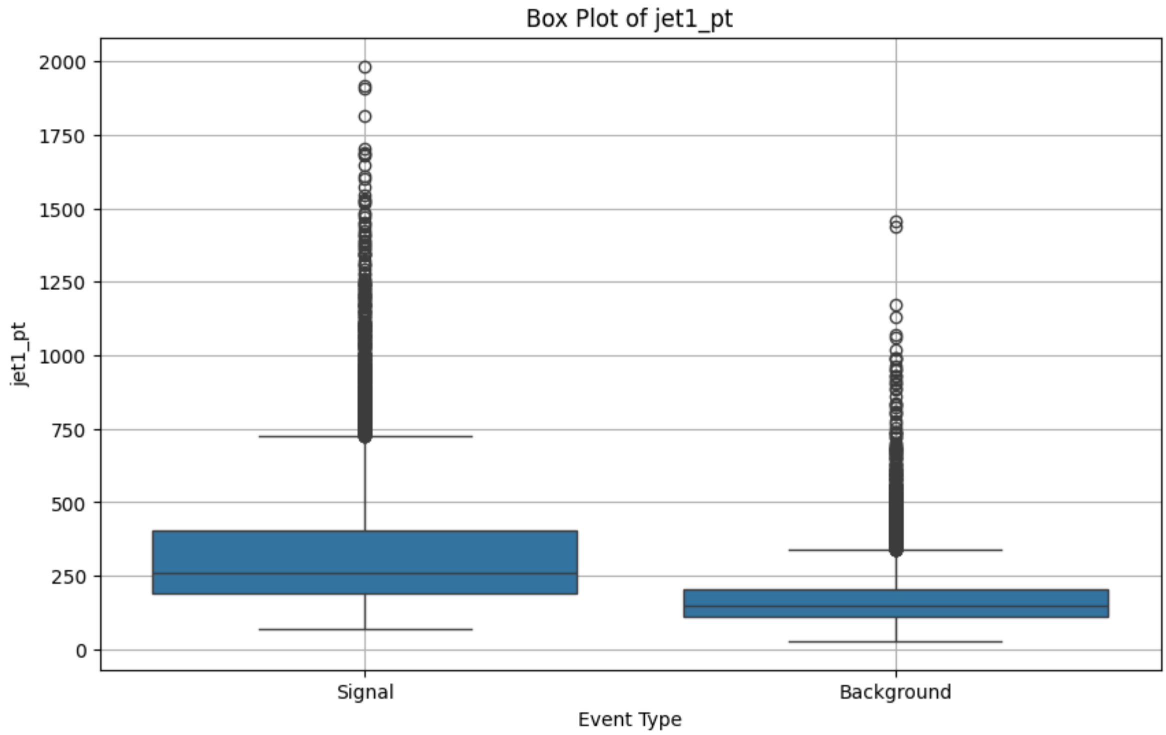
Box Plot

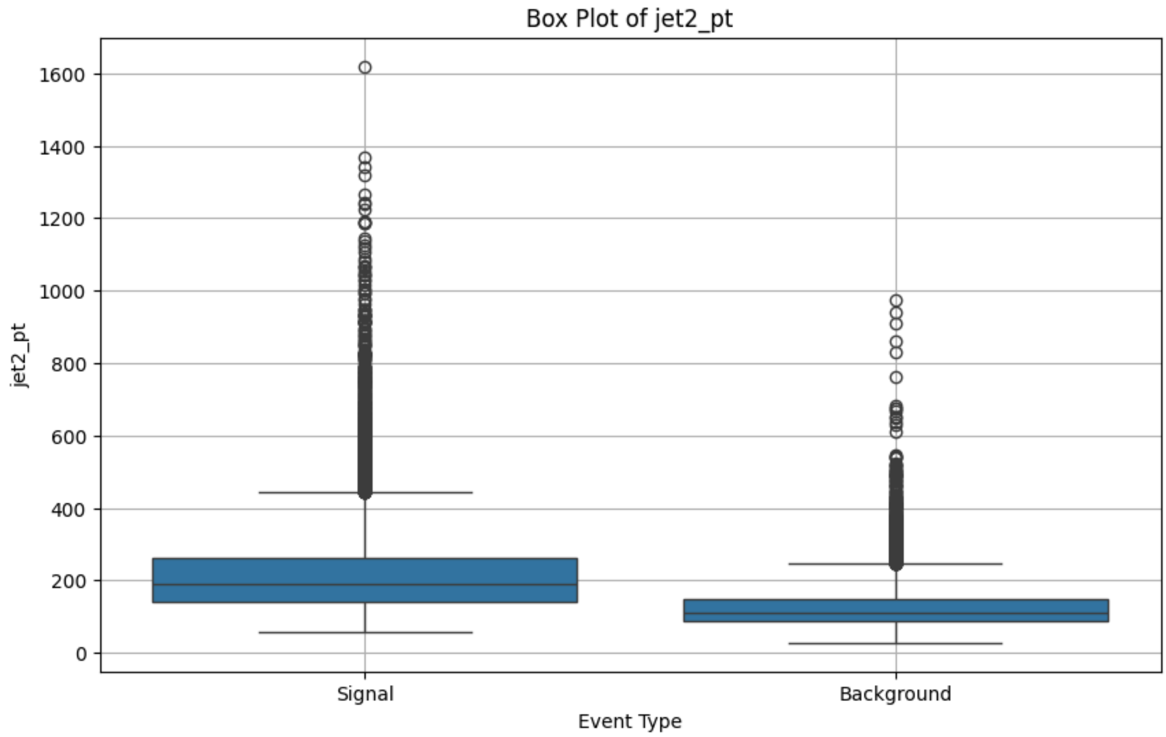
In this section, we use box plots to visualize the distributions of selected variables for both the signal and background datasets. Box plots provide insights into the spread, central tendency, and presence of outliers for each variable. This comparison can help identify variables with significant differences between the signal and background, which may be useful for classification or feature selection.

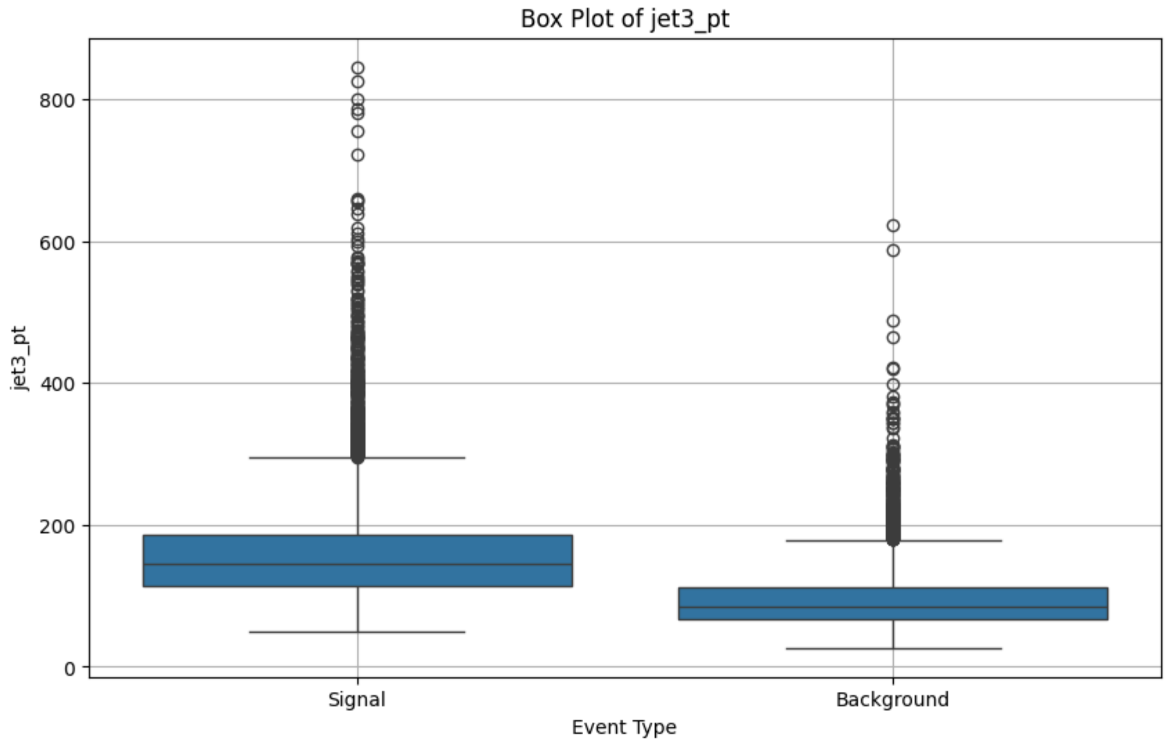
In [2]:

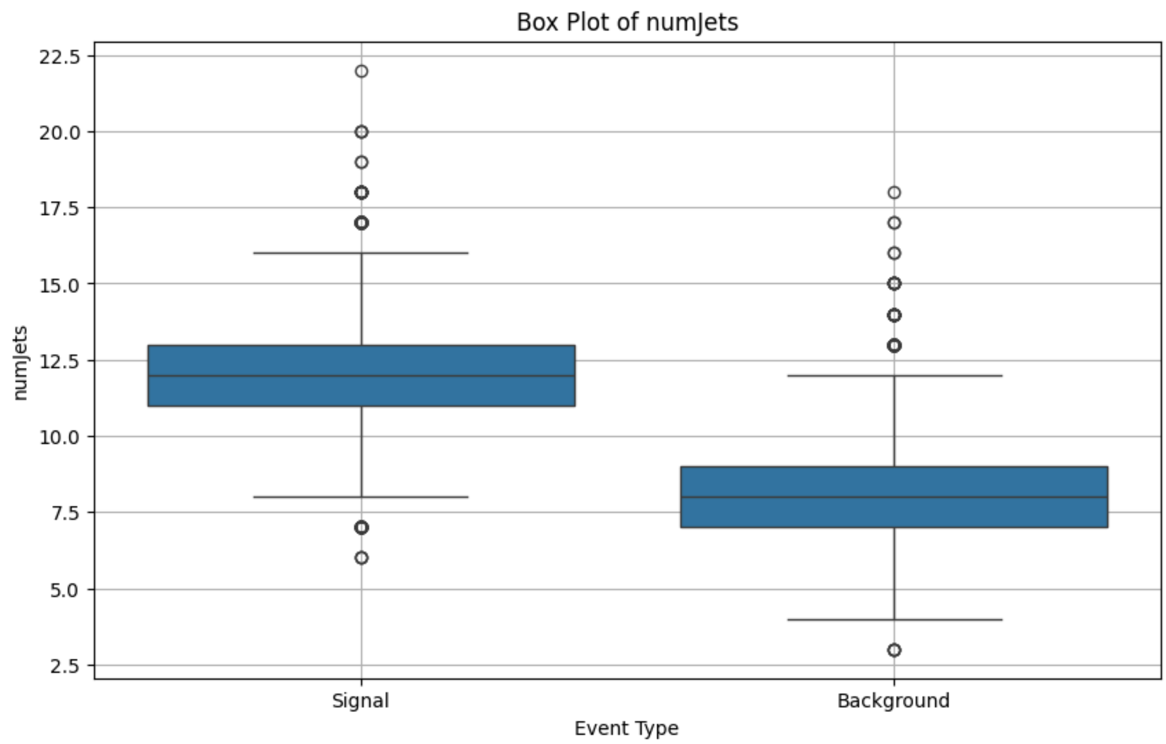
```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load your data
signal_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/SignalTTr
background_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/HBack


# Combine the data into a single DataFrame for easier plotting
signal_df['Type'] = 'Signal'
```

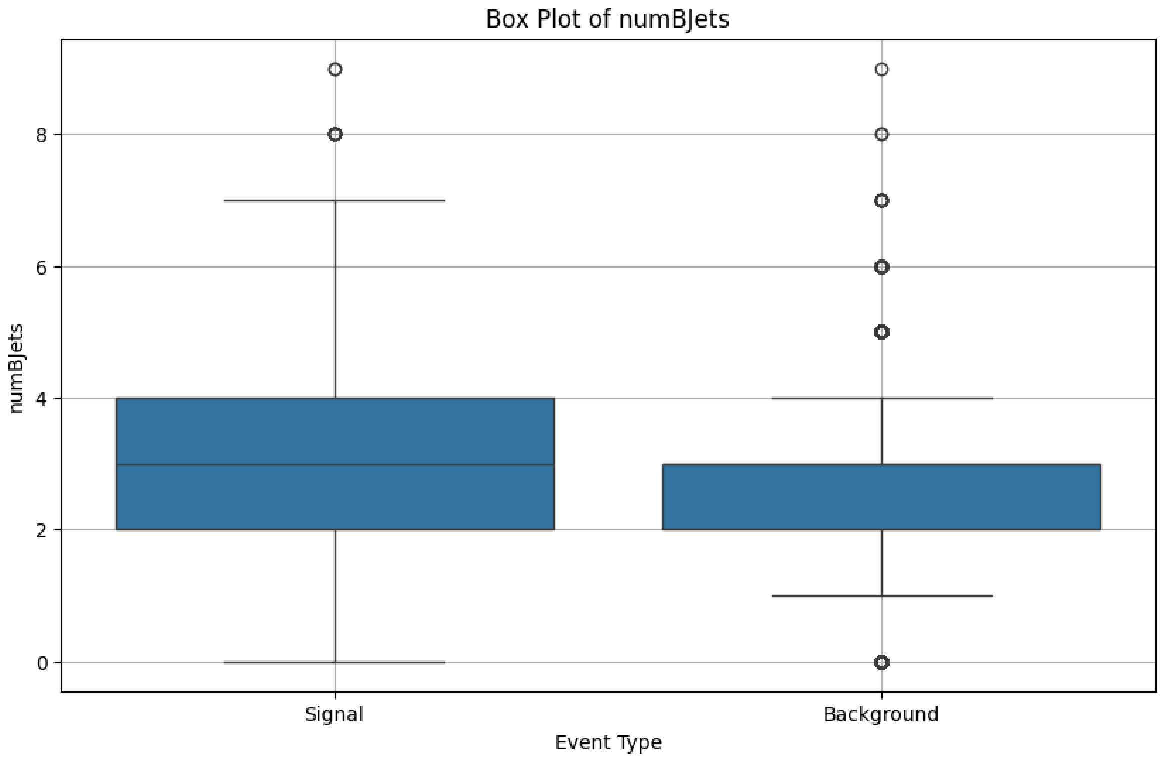
```
background_df['Type'] = 'Background'
combined_df = pd.concat([signal_df, background_df], ignore_index=True)

# List of variables to analyze
variables = ['jet1_pt', 'jet2_pt', 'jet3_pt', 'numJets', 'numBJets',
             'bjet1_pt', 'sumbjet_pt', 'Scalar_HT', 'missing_ET',
             'rat_MET_HT', 'deltaEta_jets', 'deltaPhi_jets',
             'deltaEta_bjets', 'deltaPhi_bjets', 'deltaR_jets',
             'deltaR_bjets', 'top_mass']

# Create a box plot for each variable
for var in variables:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x='Type', y=var, data=combined_df)
    plt.title(f'Box Plot of {var}')
    plt.xlabel('Event Type')
    plt.ylabel(var)
    plt.grid()
    plt.savefig(f'boxplot_{var}.png')  # Save the plot as a PNG file
    plt.show()

# Summary statistics for each variable
summary_statistics = []

for var in variables:
    signal_stats = signal_df[var].describe()
    background_stats = background_df[var].describe()
    summary_statistics.append({
        'Variable': var,
        'Signal Mean': signal_stats['mean'],
        'Signal Std Dev': signal_stats['std'],
        'Background Mean': background_stats['mean'],
        'Background Std Dev': background_stats['std'],
        'Signal IQR': signal_stats['75%'] - signal_stats['25%'],
        'Background IQR': background_stats['75%'] - background_stats['25%
    })

# Create a summary DataFrame
summary_df = pd.DataFrame(summary_statistics)

# Display the summary
print(summary_df)

# Optionally, save the summary to a CSV file
summary_df.to_csv('boxplot_summary_statistics.csv', index=False)
```

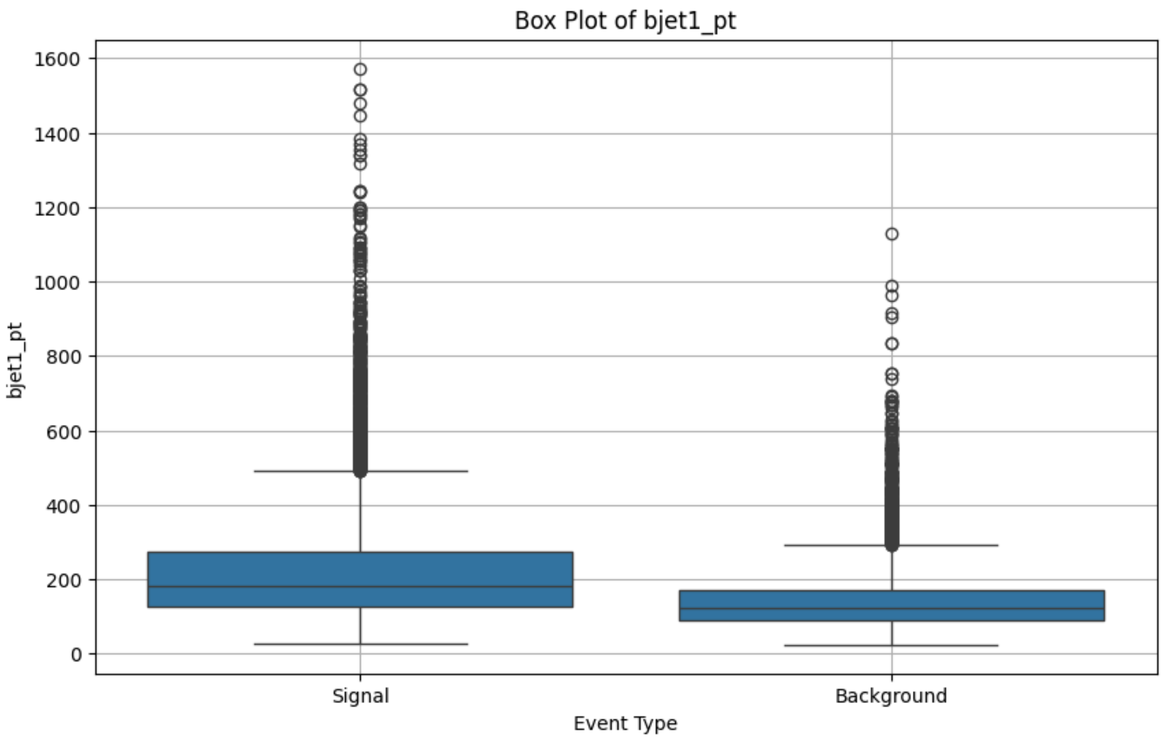

Box Plot of jet1_pt
jet1_pt
2000
1750
1500
1250
1000
750
500
250
0
Signal
Background
Event Type

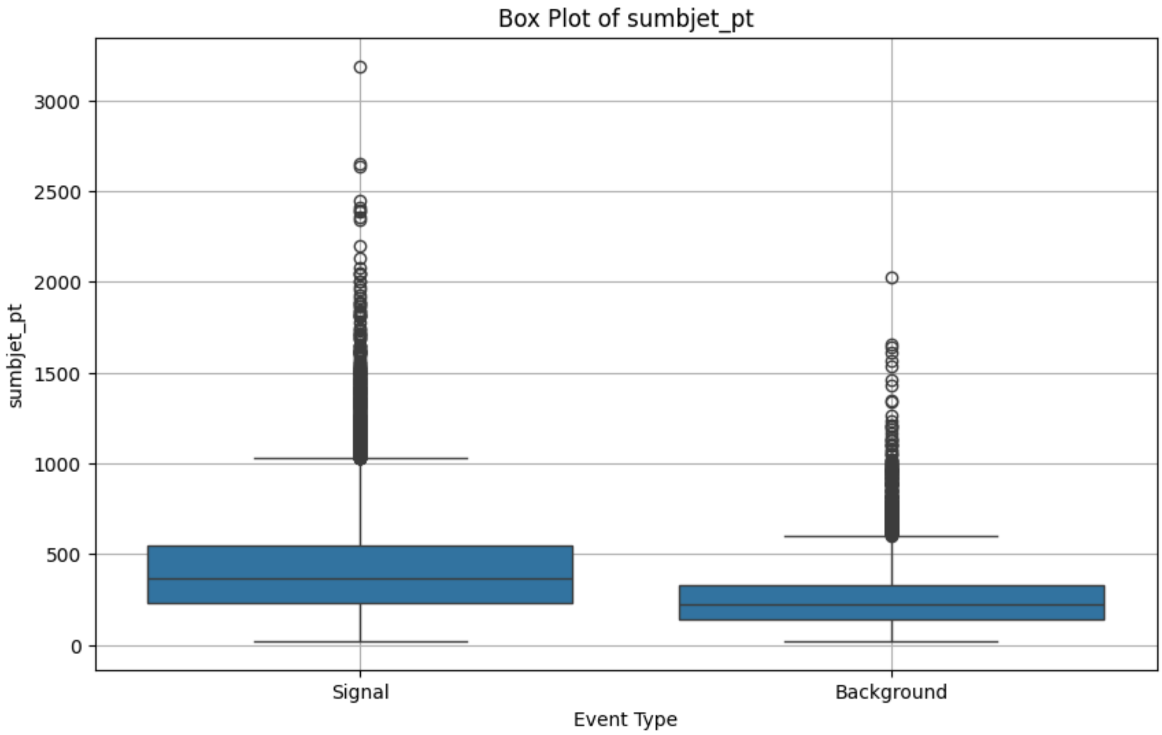

Box Plot of jet2_pt
jet2_pt
1600
1400
1200
1000
800
600
400
200
0
Signal
Background
Event Type

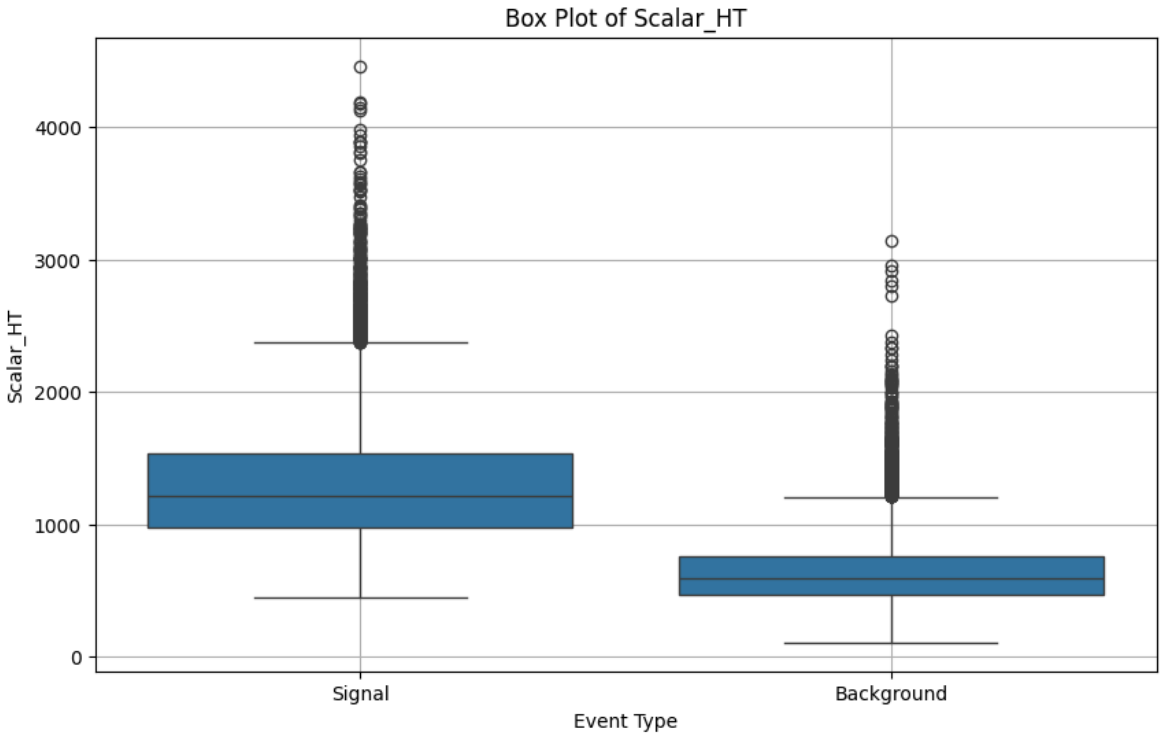

Box Plot of jet3_pt
jet3_pt
0
200
400
600
800
Signal
Background
Event Type

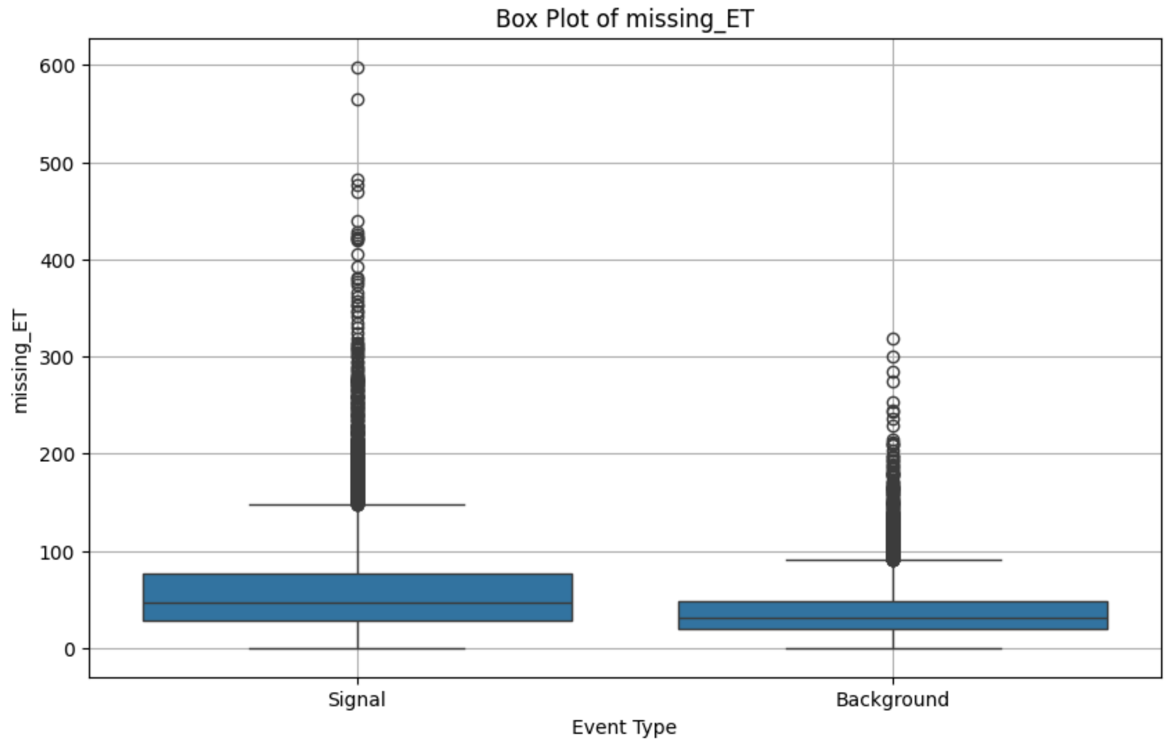

Box Plot of numJets
numJets
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
Signal
Background
Event Type

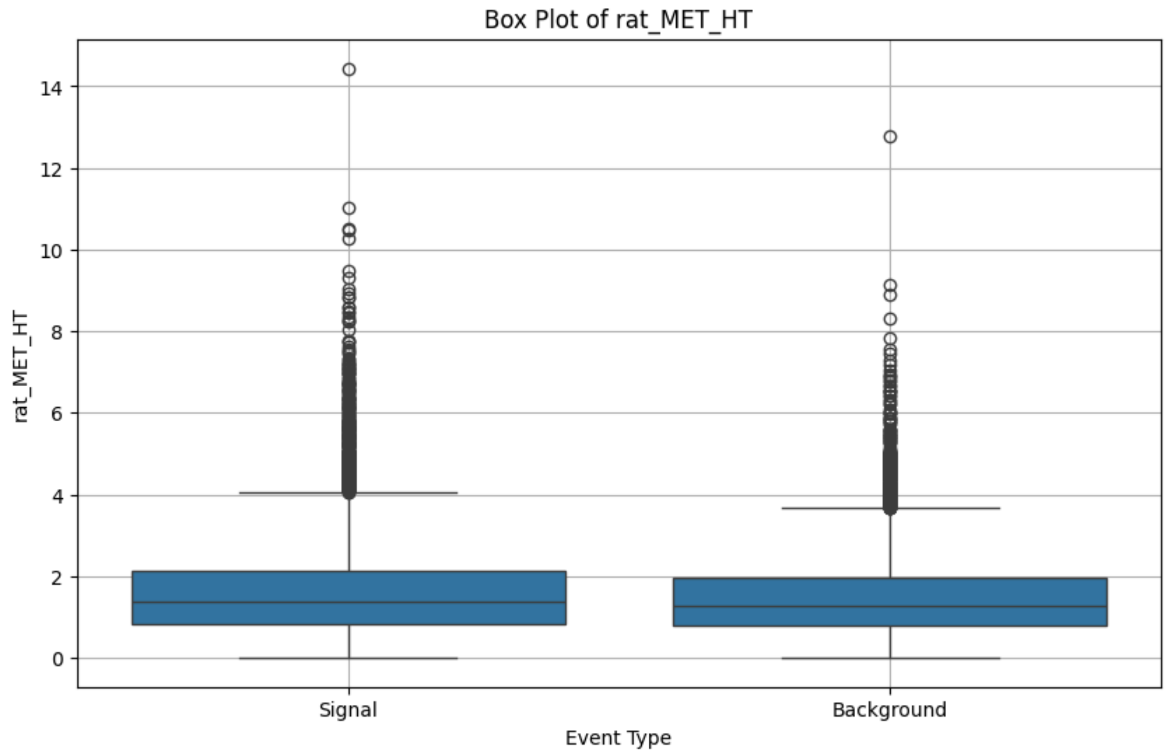

Box Plot of numBJets
numBJets
0
2
4
6
8
Signal
Background
Event Type

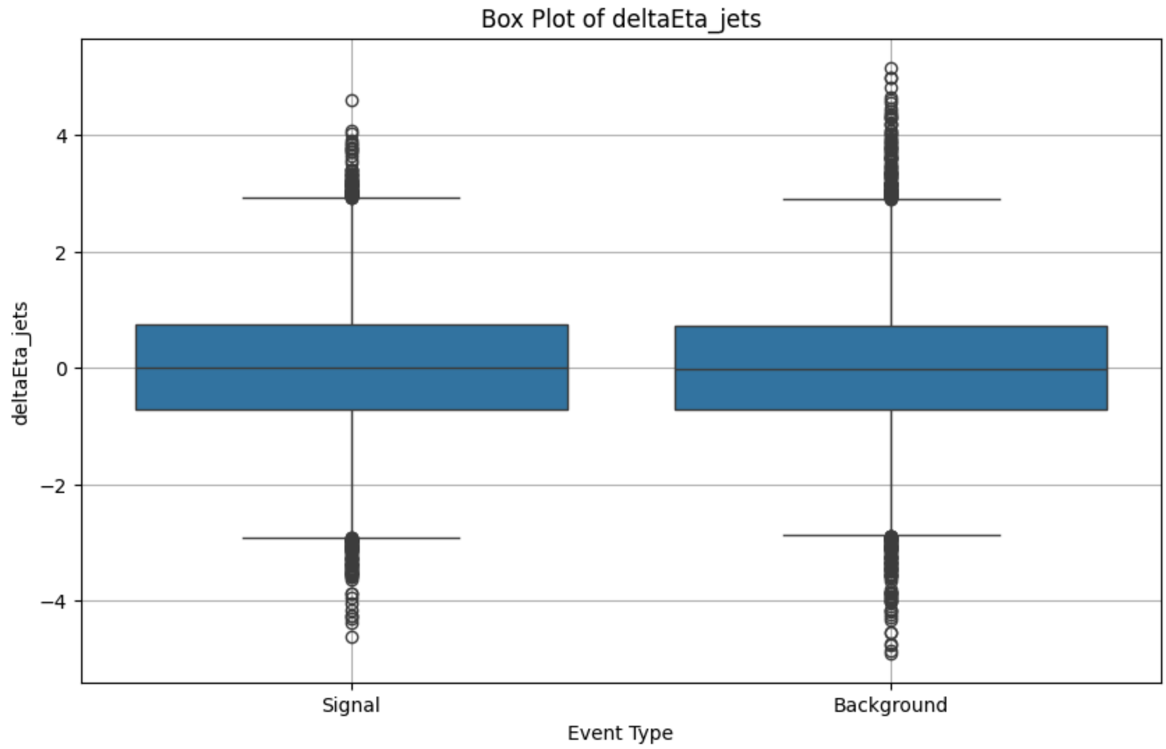

Box Plot of bjet1_pt
bjet1_pt
0
200
400
600
800
1000
1200
1400
1600
Signal
Background
Event Type

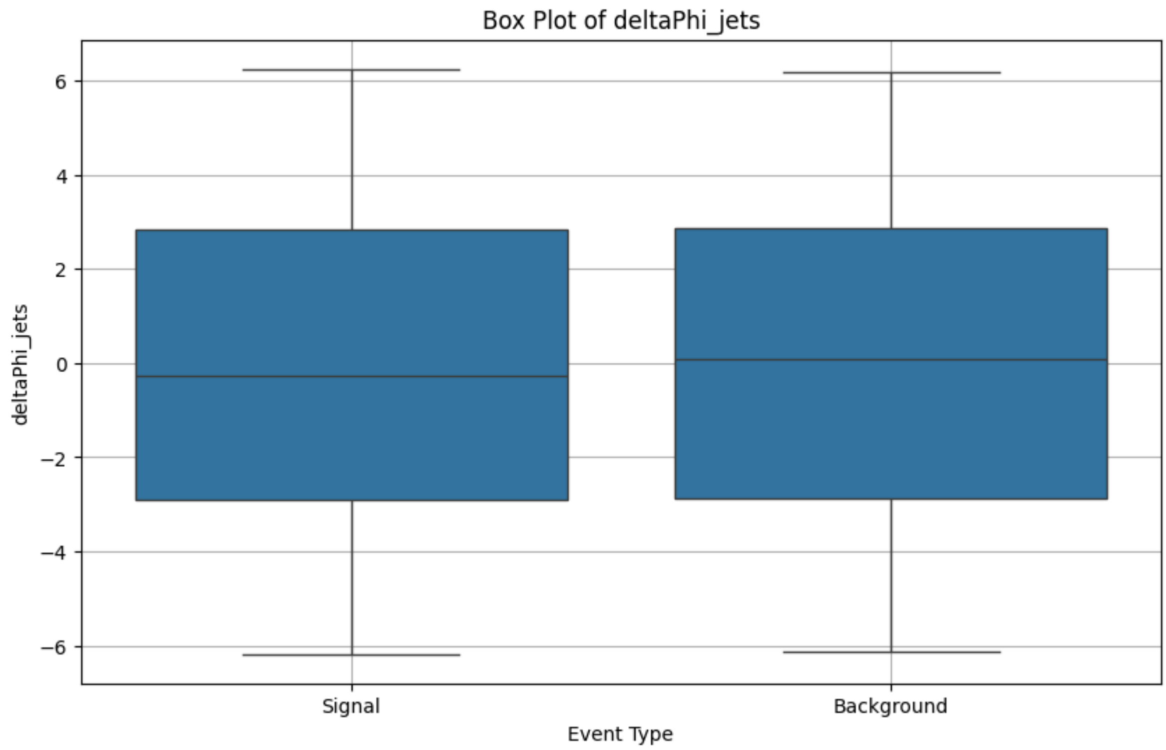

Box Plot of sumbjet_pt
3000
2500
2000
1500
1000
500
0
sumbjet_pt
Signal
Background
Event Type

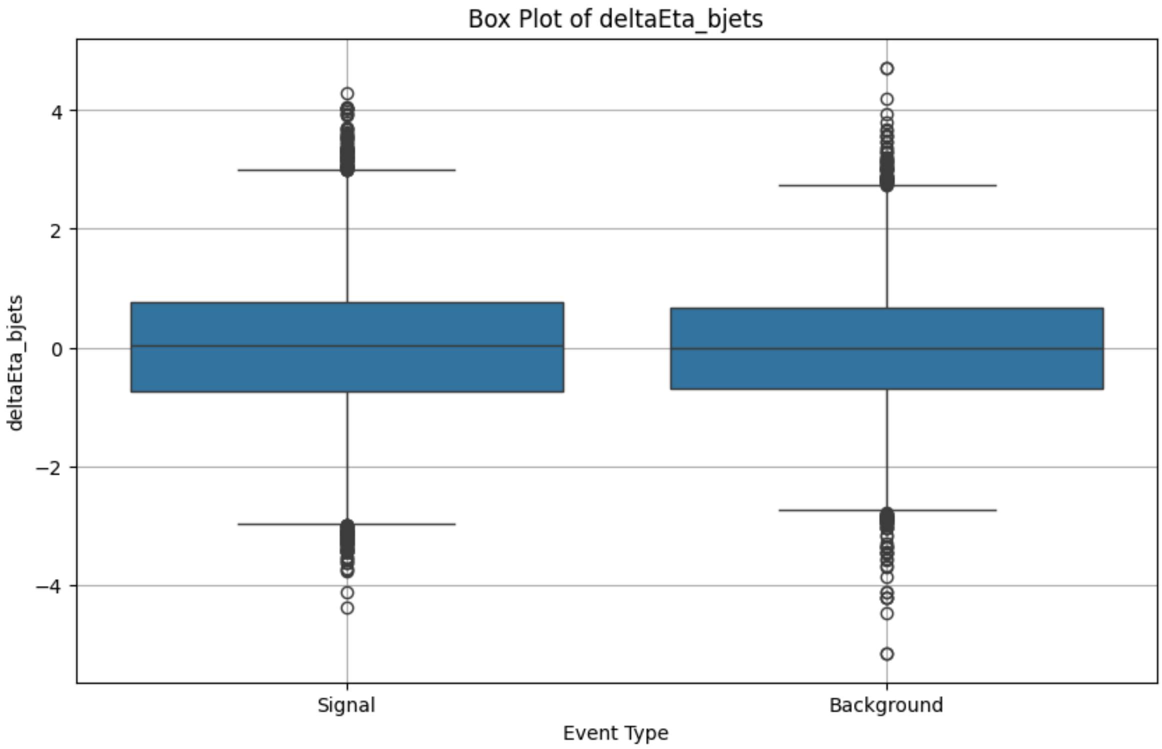

Box Plot of Scalar_HT
4000
3000
2000
1000
0
Scalar_HT
Signal
Background
Event Type

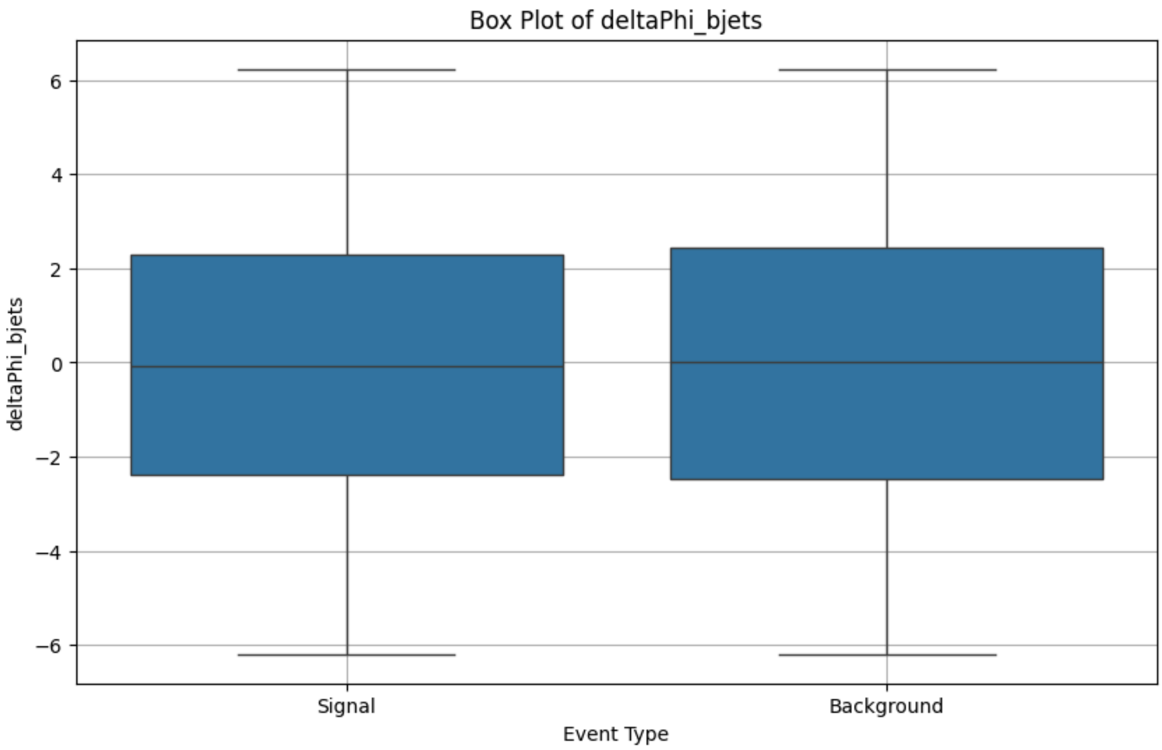

Box Plot of missing_ET
missing_ET
0
100
200
300
400
500
600
Signal
Background
Event Type

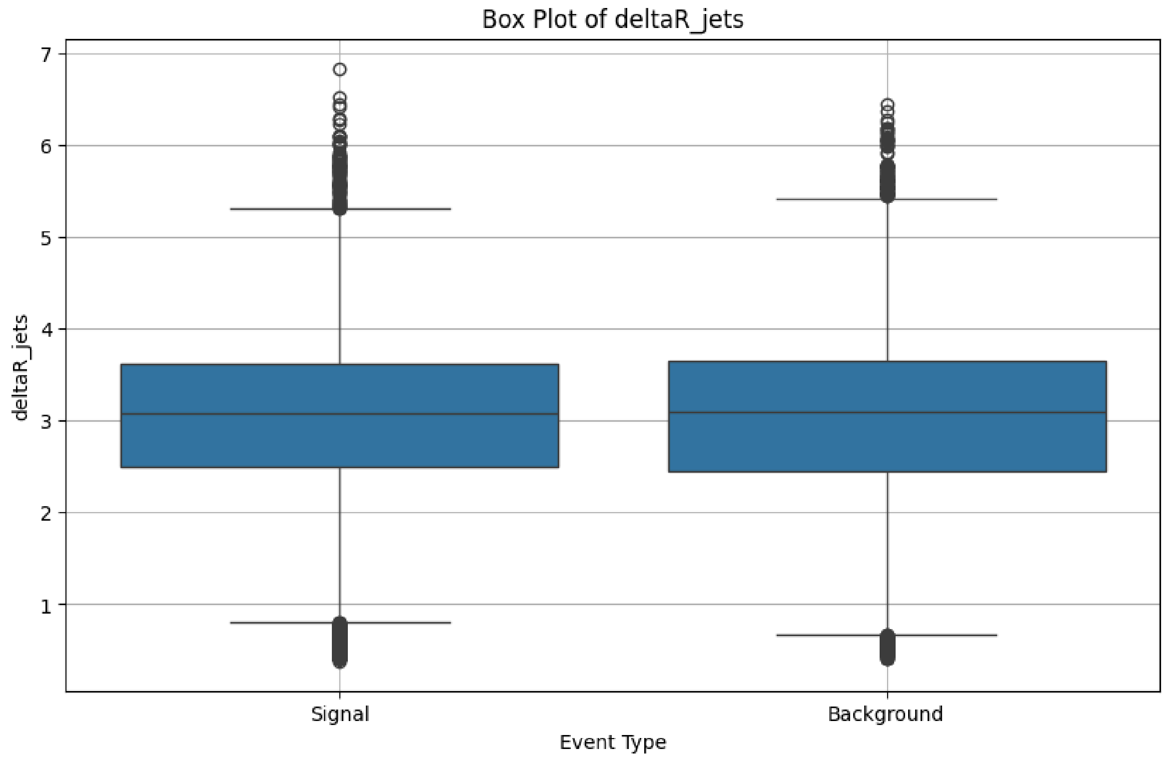

Box Plot of rat_MET_HT
rat_MET_HT
0
2
4
6
8
10
12
14
Signal
Background
Event Type

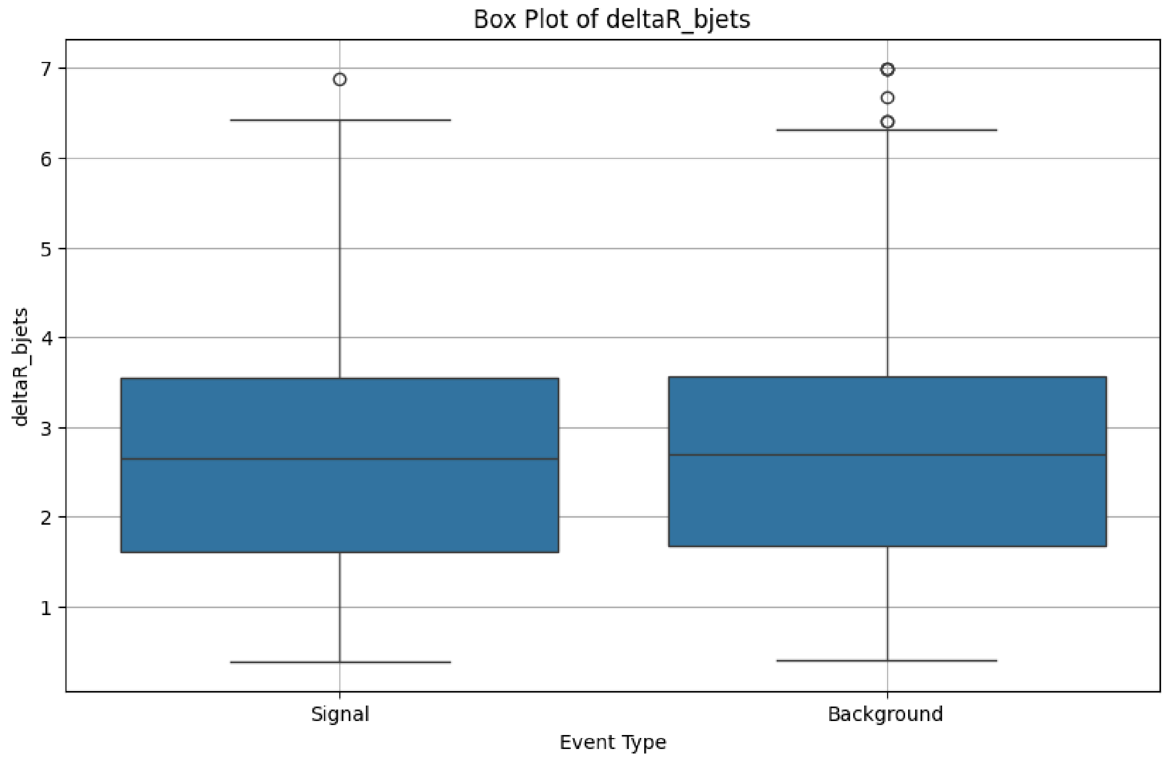

Box Plot of deltaEta_jets
4
2
0
−2
−4
deltaEta_jets
Signal
Background
Event Type

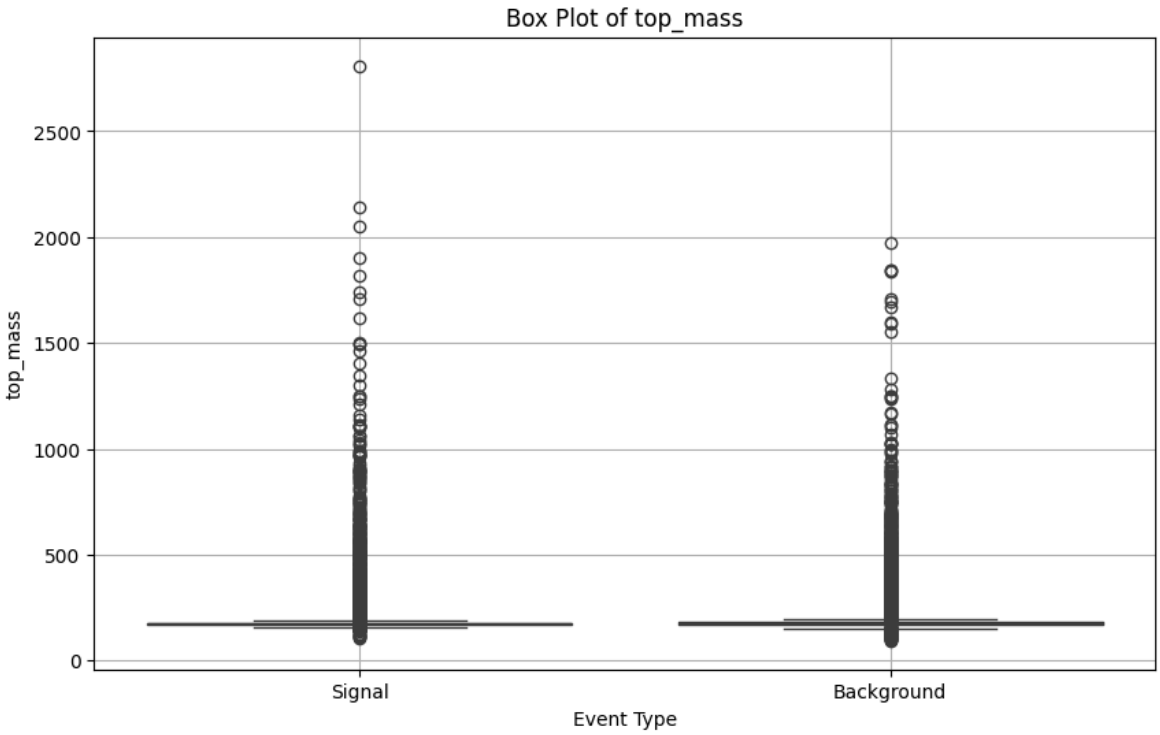

Box Plot of deltaPhi_jets
6
4
2
0
−2
−4
−6
deltaPhi_jets
Signal
Background
Event Type


Box Plot of deltaEta_bjets
deltaEta_bjets
4
2
0
−2
−4
Signal
Background
Event Type


Box Plot of deltaPhi_bjets
deltaPhi_bjets
6
4
2
0
−2
−4
−6
Signal
Background
Event Type


Box Plot of deltaR_jets
deltaR_jets
Signal
Background
Event Type


Box Plot of deltaR_bjets
deltaR_bjets
Signal
Background
Event Type



	Variable	Signal Mean	Signal Std Dev	Background Mean	\
0	jet1_pt	327.913806	210.909642	174.845906	
1	jet2_pt	227.726548	138.746388	128.249269	
2	jet3_pt	160.075479	70.473132	95.247804	
3	numJets	11.907100	1.867784	7.992100	
4	numBJets	3.220000	1.346618	2.628500	
5	bjet1_pt	228.028349	160.211425	141.447562	
6	sumbjet_pt	431.375690	293.190688	255.092182	
7	Scalar_HT	1307.858939	471.347441	648.850664	
8	missing_ET	59.393203	48.249683	37.499789	
9	rat_MET_HT	1.635210	1.181753	1.487517	
10	deltaEta_jets	0.001854	1.149845	-0.001165	
11	deltaPhi_jets	-0.052621	2.938624	0.007301	
12	deltaEta_bjets	0.014526	1.178438	-0.008204	
13	deltaPhi_bjets	-0.065385	2.699535	0.009292	
14	deltaR_jets	2.990983	1.006753	3.015709	
15	deltaR_bjets	2.641749	1.304285	2.691509	
16	top_mass	188.448369	99.765615	195.010709	

	Background Std Dev	Signal IQR	Background IQR
0	100.716293	214.829422	90.726904
1	66.860378	120.172493	63.499533
2	41.381533	72.788029	44.550486
3	1.695501	2.000000	2.000000
4	1.188036	2.000000	1.000000
5	82.622212	145.978680	81.873920
6	165.144226	318.522884	184.454403
7	269.878435	561.555435	297.686256
8	27.502012	47.863606	28.774477
9	0.981800	1.293434	1.151557
10	1.178140	1.462717	1.444782
11	2.952025	5.749130	5.753300
12	1.045917	1.494457	1.374553
13	2.786250	4.685269	4.927325
14	1.003550	1.126661	1.191551
15	1.269779	1.943626	1.877583
16	101.202783	7.724556	11.058369

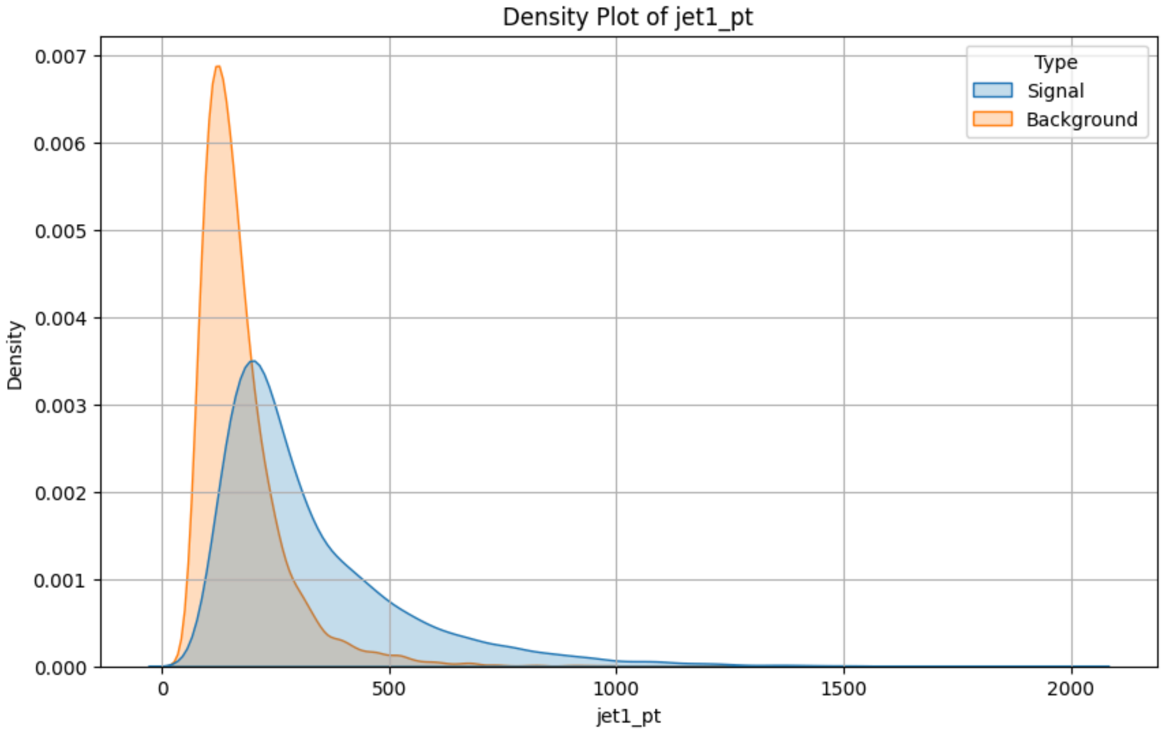
Density Plot

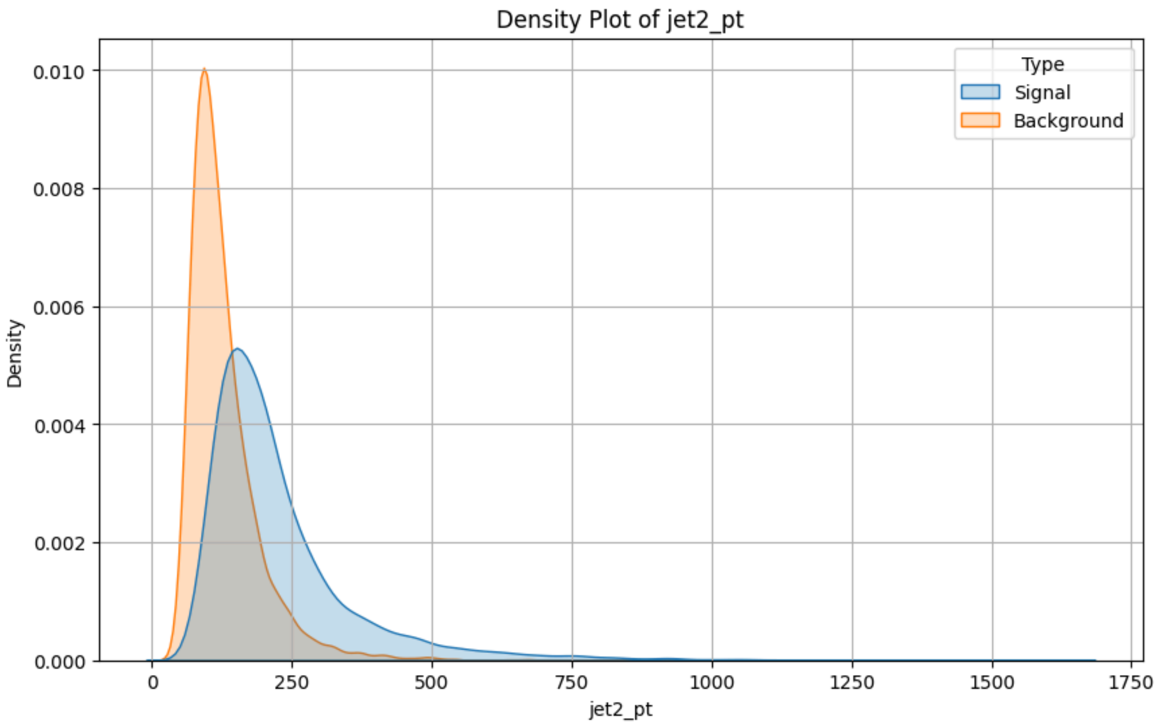
In this section, we visualize the distributions of selected variables for both the signal and background datasets using density plots. Density plots provide a smoothed representation of the distribution, allowing us to observe the underlying patterns and differences between the signal and background. This visualization is particularly useful for understanding the shapes of the distributions and identifying overlapping regions, which can inform classification tasks and feature selection.

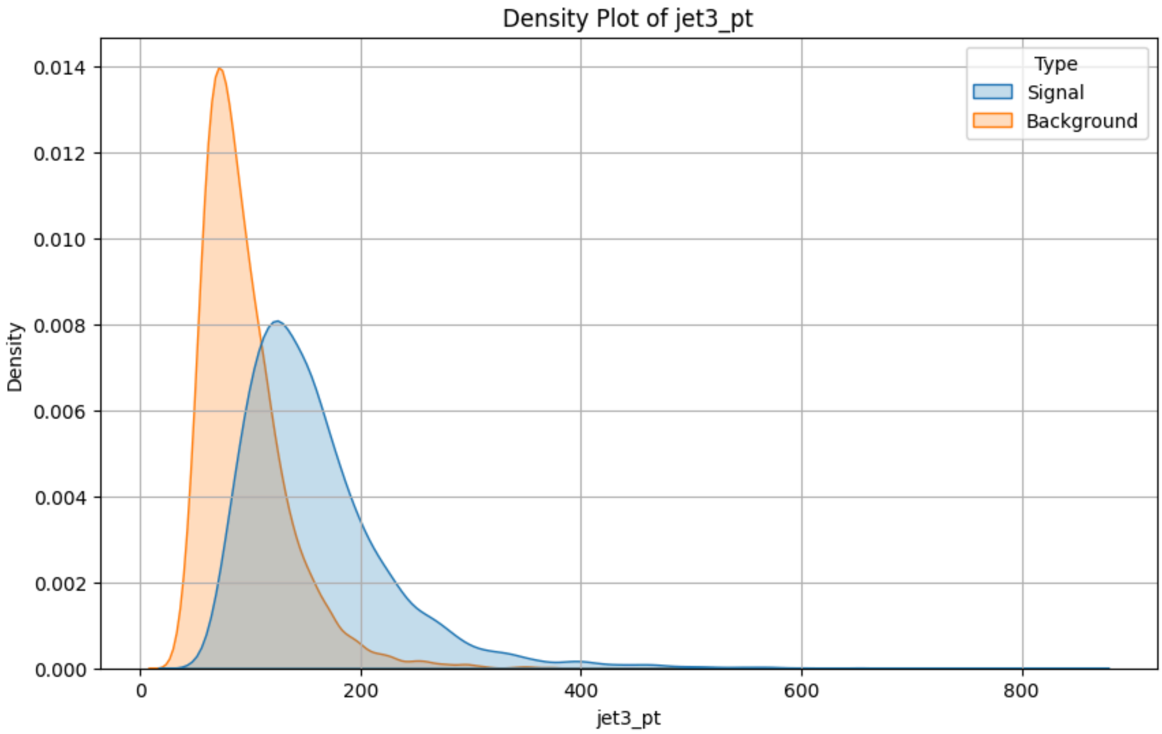
In [3]:

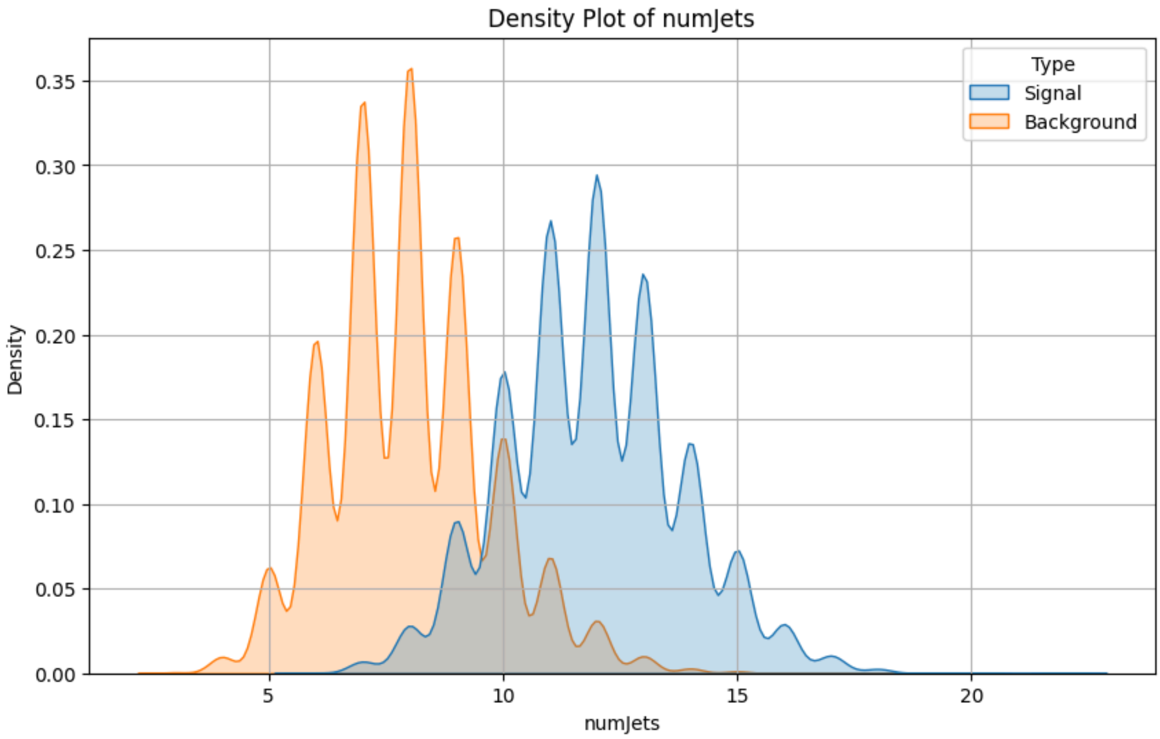
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load your data
signal_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/SignalTTr
background_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/HBack

# Combine the data into a single DataFrame for easier plotting
signal_df['Type'] = 'Signal'
background_df['Type'] = 'Background'
combined_df = pd.concat([signal_df, background_df], ignore_index=True)

# List of variables to analyze
variables = ['jet1_pt', 'jet2_pt', 'jet3_pt', 'numJets', 'numBJets',
             'bjet1_pt', 'sumbjet_pt', 'Scalar_HT', 'missing_ET',
             'rat_MET_HT', 'deltaEta_jets', 'deltaPhi_jets',
             'deltaEta_bjets', 'deltaPhi_bjets', 'deltaR_jets',
             'deltaR_bjets', 'top_mass']

# Create density plots for each variable
for var in variables:
    plt.figure(figsize=(10, 6))
    sns.kdeplot(data=combined_df, x=var, hue='Type', fill=True, common_no
    plt.title(f'Density Plot of {var}')
    plt.xlabel(var)
    plt.ylabel('Density')
    plt.grid()
    plt.savefig(f'density_plot_{var}.png')  # Save the plot as a PNG file
    plt.show()

# Summary statistics for each variable
summary_statistics = []

for var in variables:
    signal_stats = signal_df[var].describe()
    background_stats = background_df[var].describe()
    summary_statistics.append({
        'Variable': var,
        'Signal Mean': signal_stats['mean'],
        'Signal Std Dev': signal_stats['std'],
        'Background Mean': background_stats['mean'],
        'Background Std Dev': background_stats['std'],
        'Signal IQR': signal_stats['75%'] - signal_stats['25%'],
        'Background IQR': background_stats['75%'] - background_stats['25%
    })
```

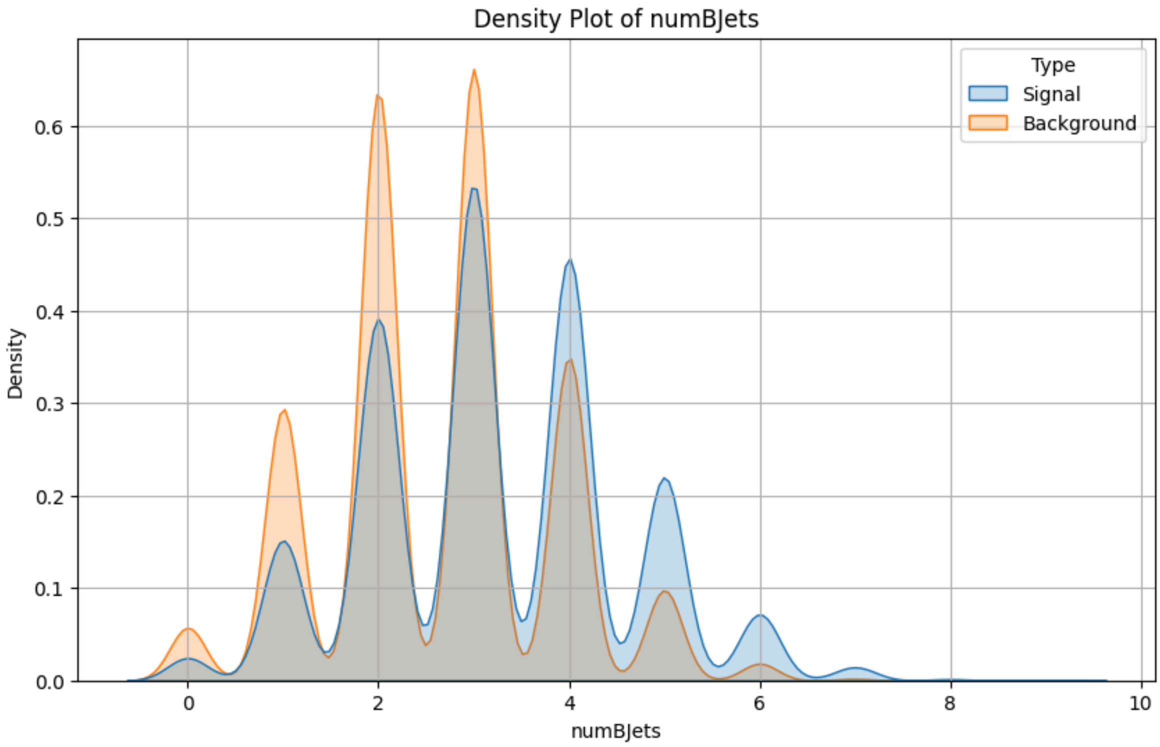
```
# Create a summary DataFrame
summary_df = pd.DataFrame(summary_statistics)

# Display the summary
print(summary_df)

# Optionally, save the summary to a CSV file
summary_df.to_csv('density_plot_summary_statistics.csv', index=False)
```

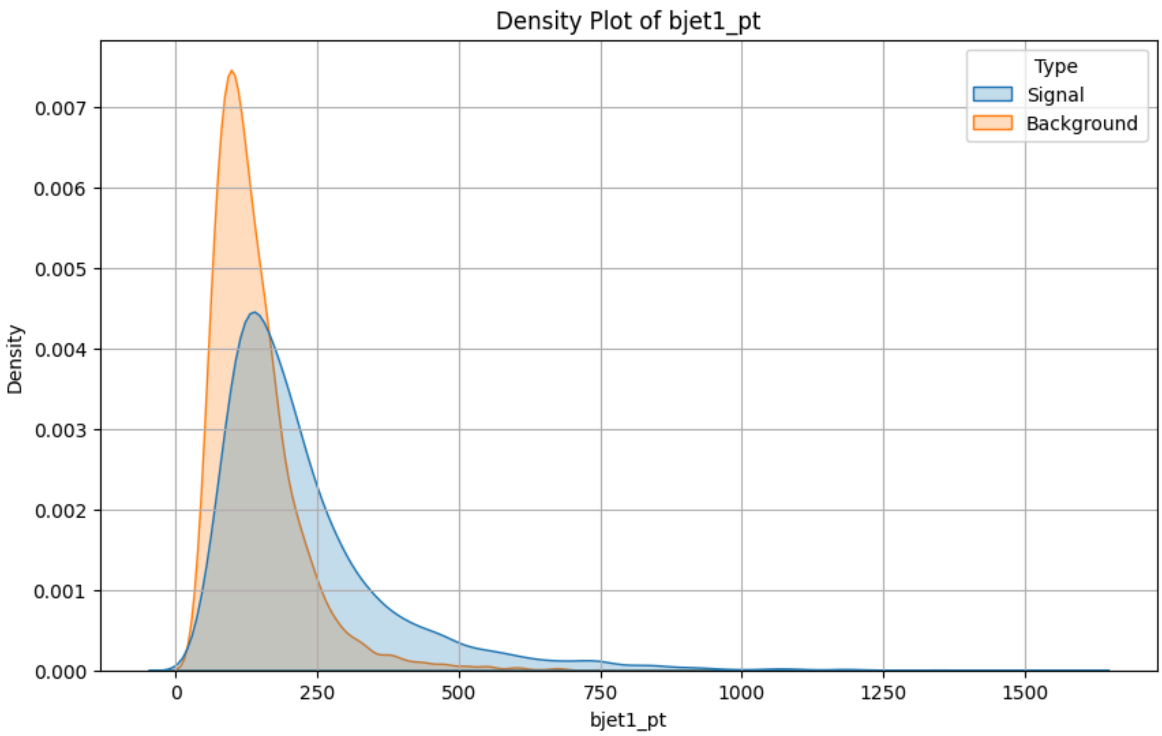


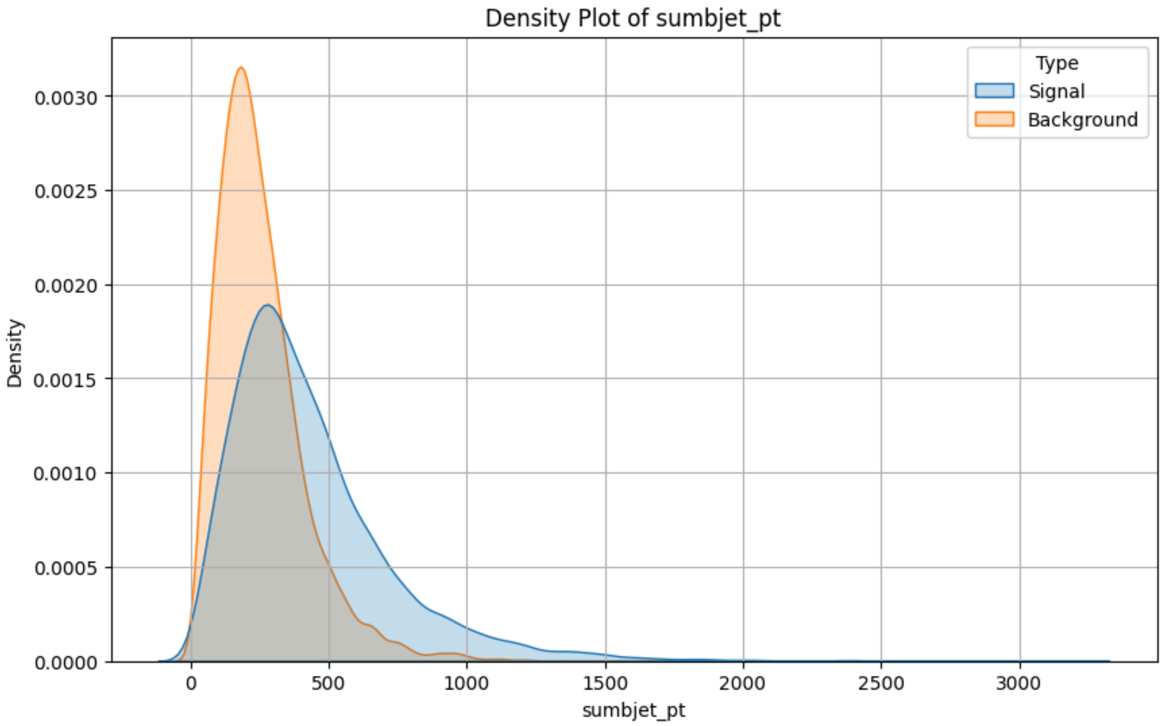


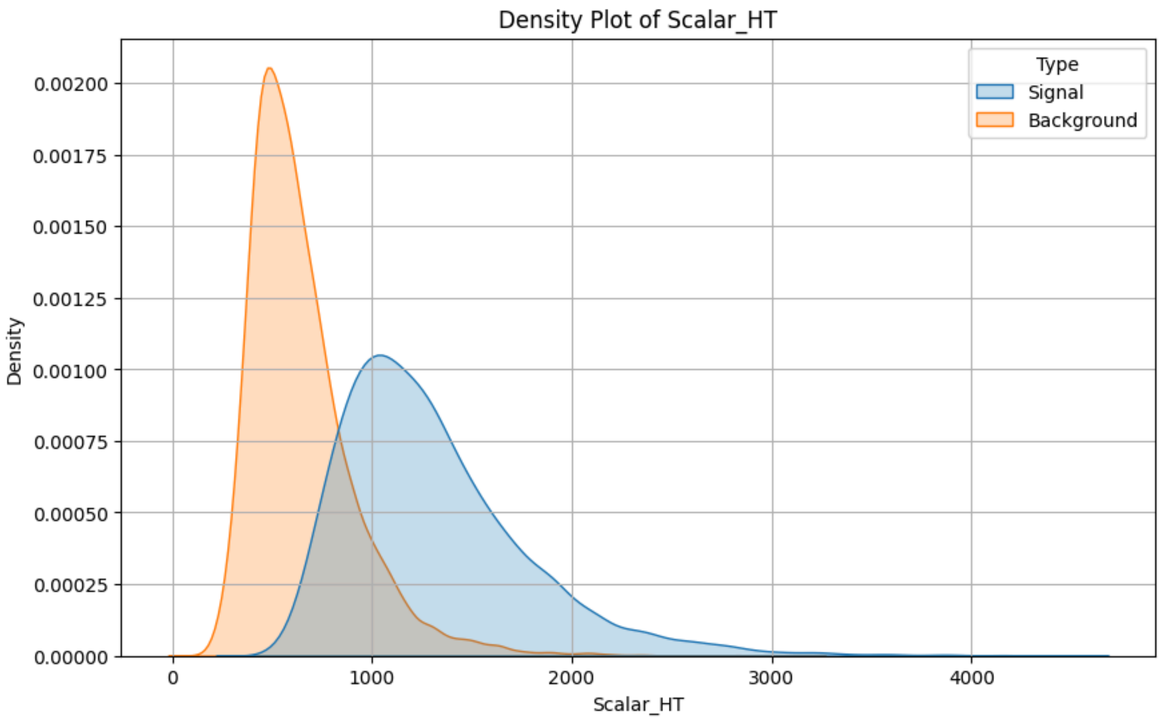

Density Plot of jet3_pt
Type
Signal
Background
Density
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
200
400
600
800
jet3_pt

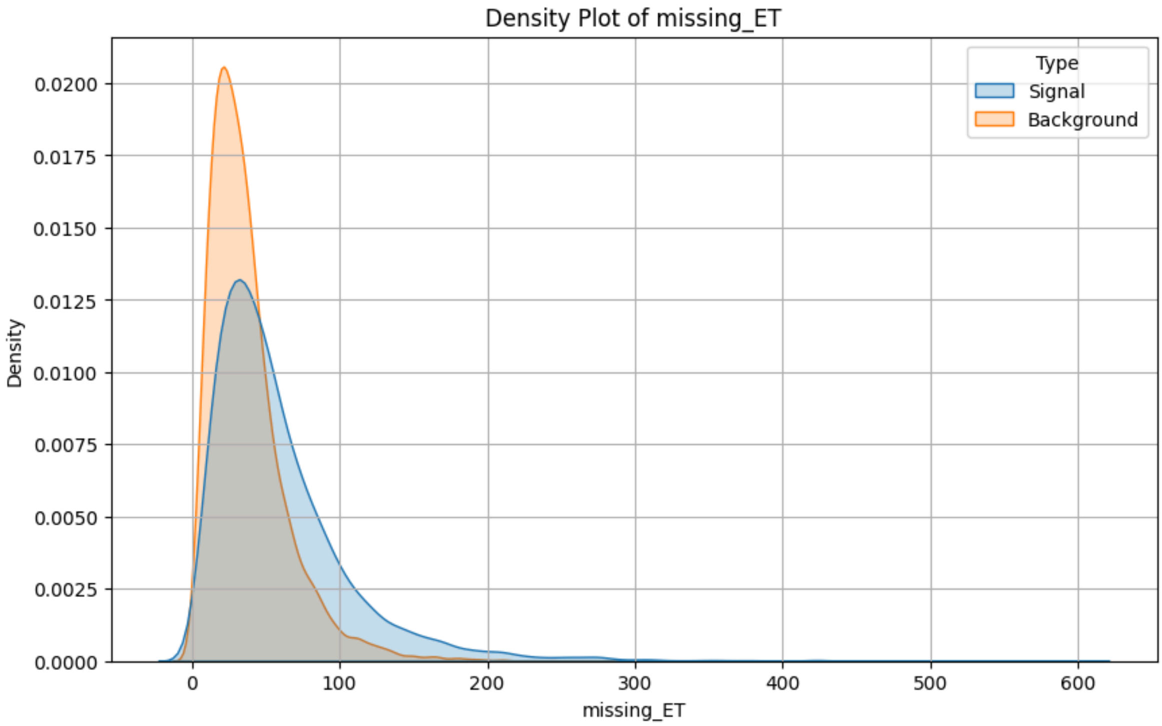

Density Plot of numJets
Type
Signal
Background
Density
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
5
10
15
20
numJets

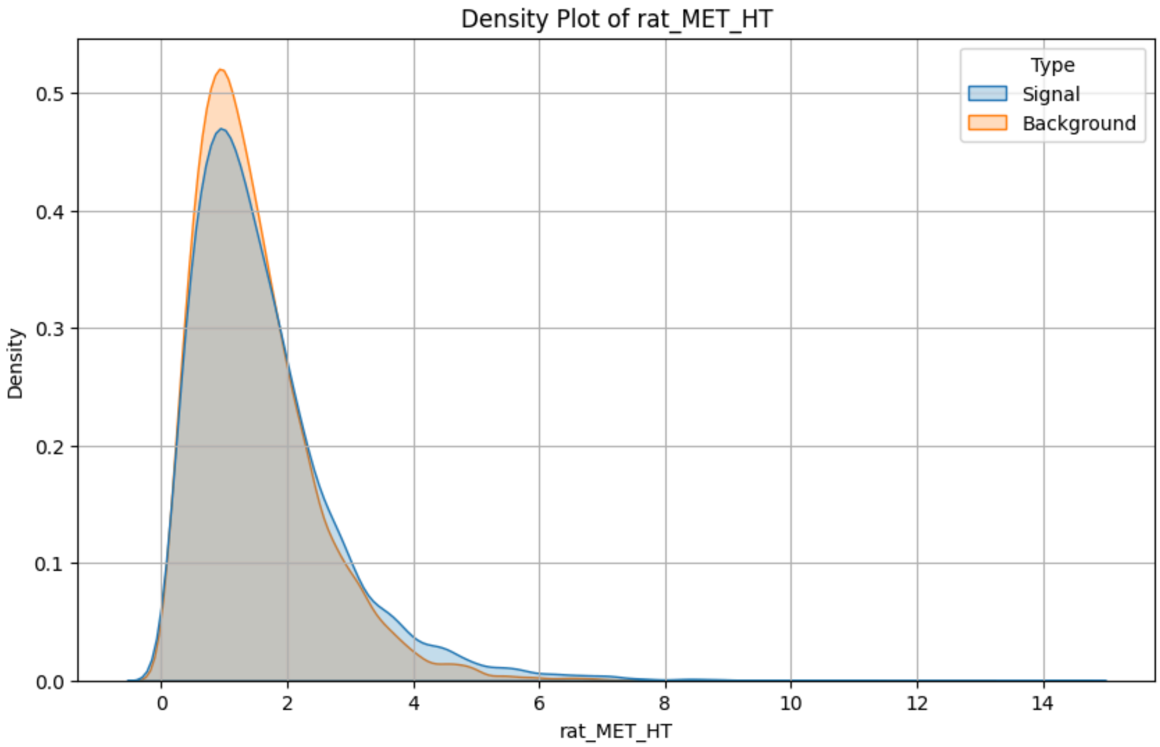

Density Plot of numBJets
Type
Signal
Background
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
2
4
6
8
10
numBJets

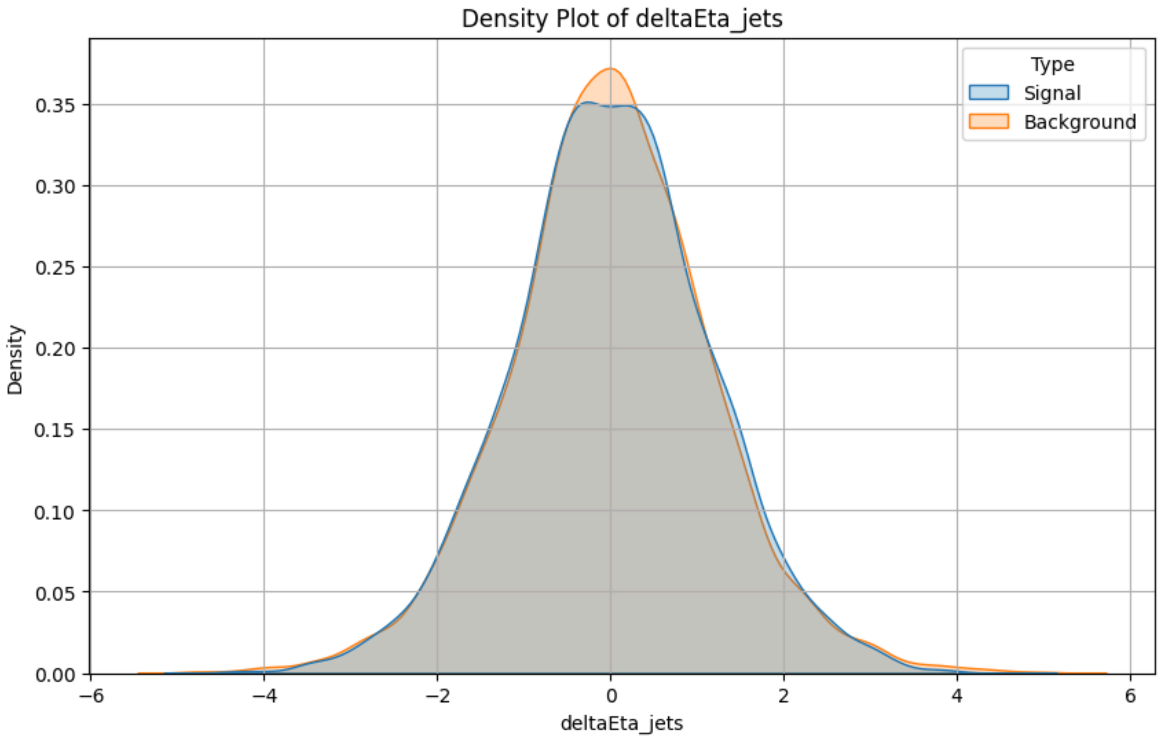

Density Plot of bjet1_pt
Type
Signal
Background
Density
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0
250
500
750
1000
1250
1500
bjet1_pt

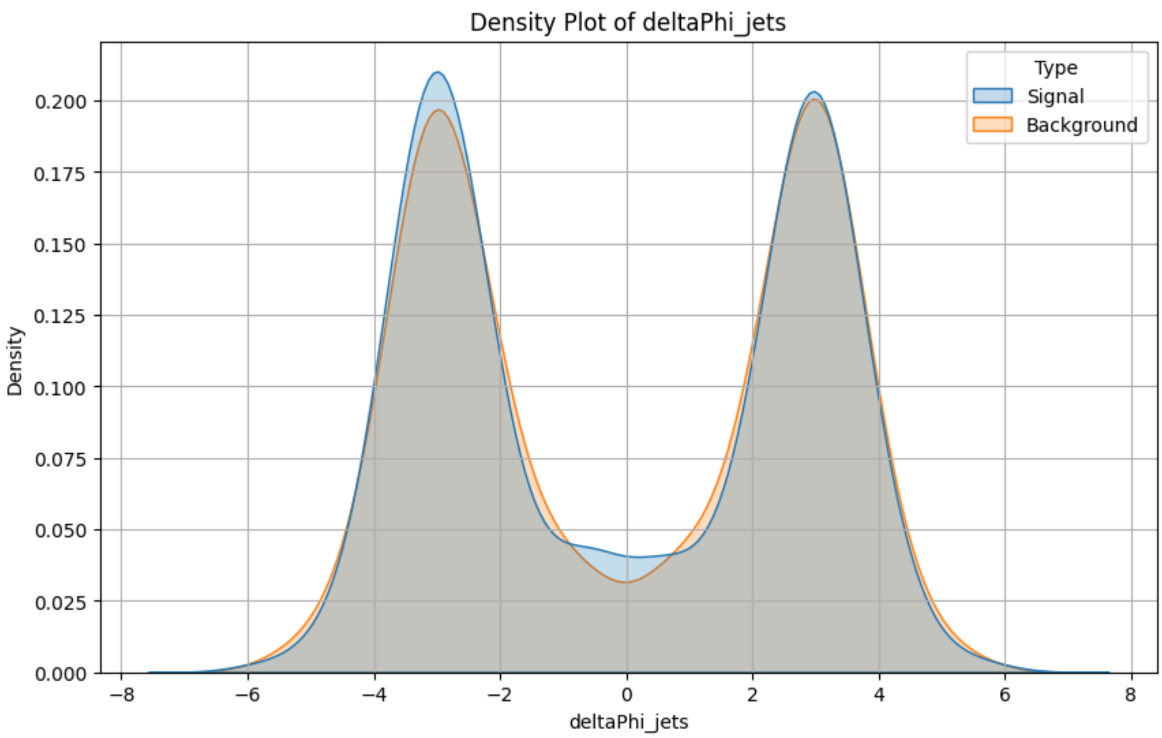

Density Plot of sumbjet_pt
Type
Signal
Background
Density
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
500
1000
1500
2000
2500
3000
sumbjet_pt

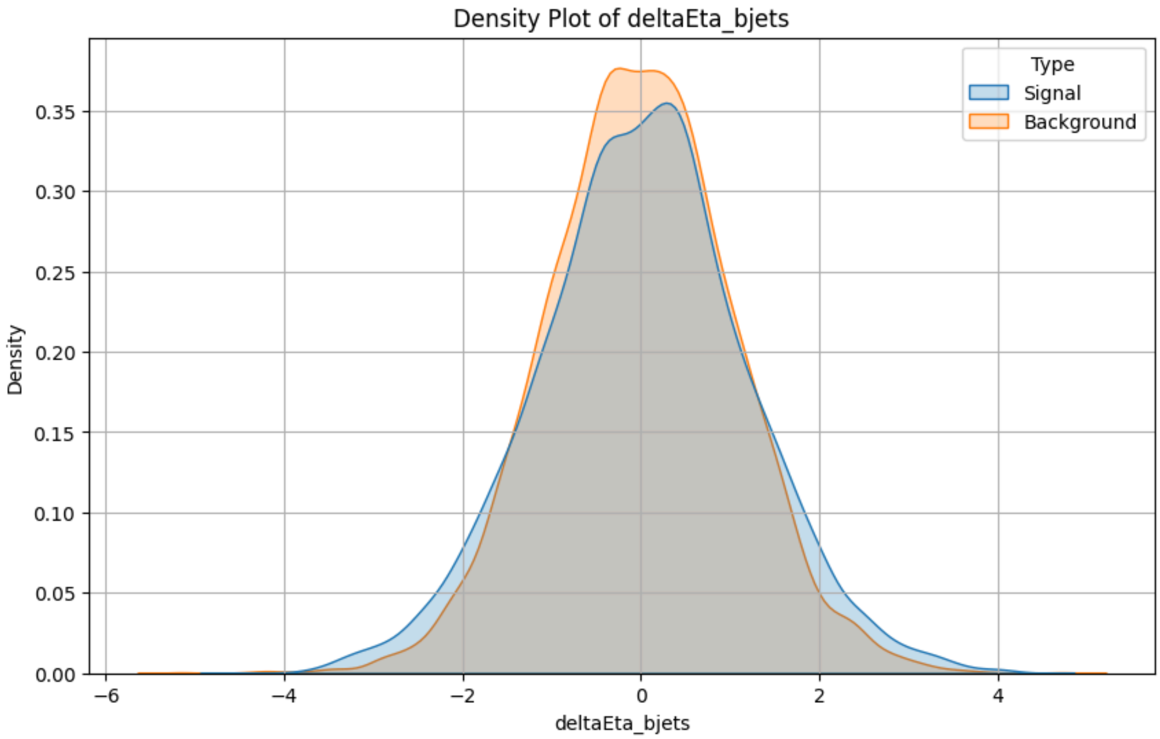

Density Plot of Scalar_HT
Type
Signal
Background
Density
0.00200
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
0
1000
2000
3000
4000
Scalar_HT


Density Plot of missing_ET
Type
Signal
Background
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
0
100
200
300
400
500
600
missing_ET

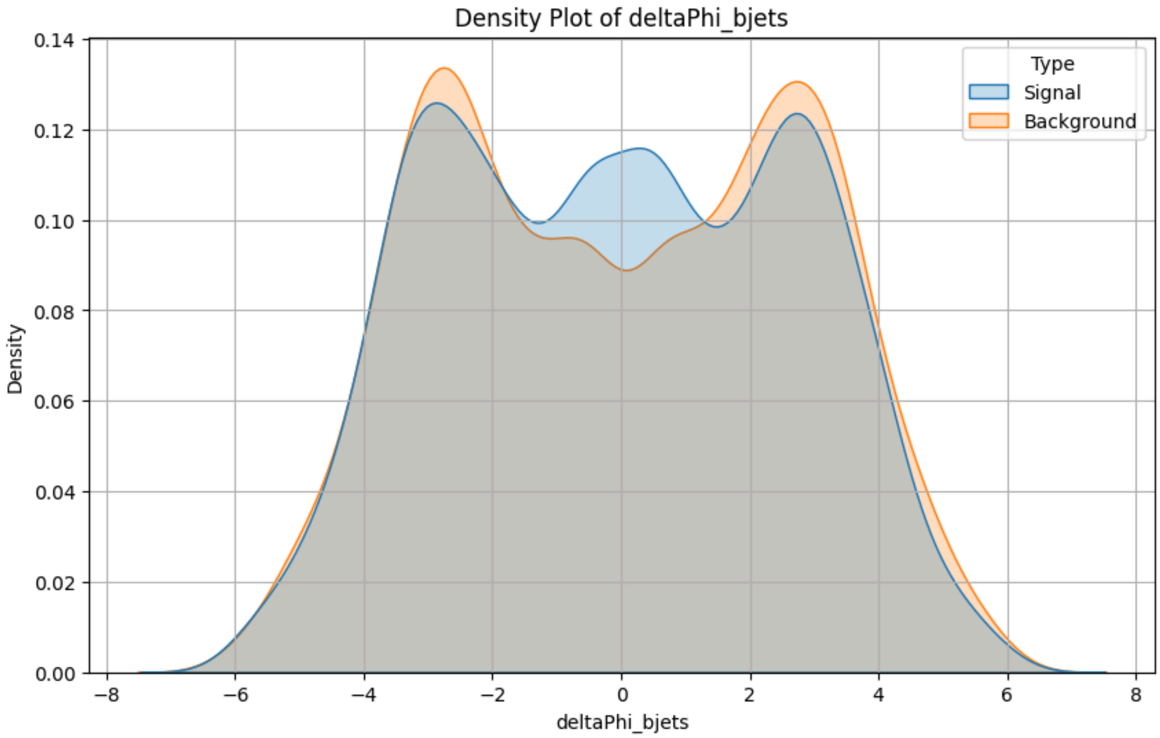

Density Plot of rat_MET_HT
Type
Signal
Background
Density
0.0
0.1
0.2
0.3
0.4
0.5
0
2
4
6
8
10
12
14
rat_MET_HT


Density Plot of deltaEta_jets
Type
Signal
Background
Density
deltaEta_jets
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
−6
−4
−2
0
2
4
6

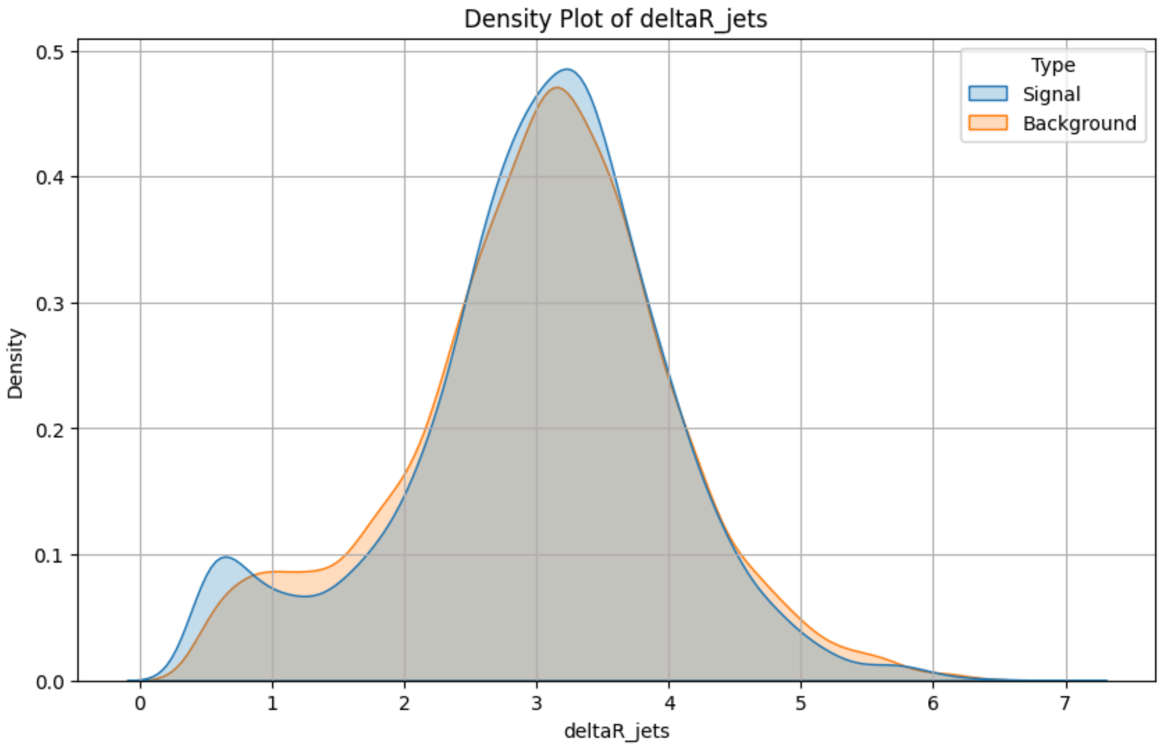

Density Plot of deltaPhi_jets
Type
Signal
Background
Density
deltaPhi_jets
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
−8
−6
−4
−2
0
2
4
6
8

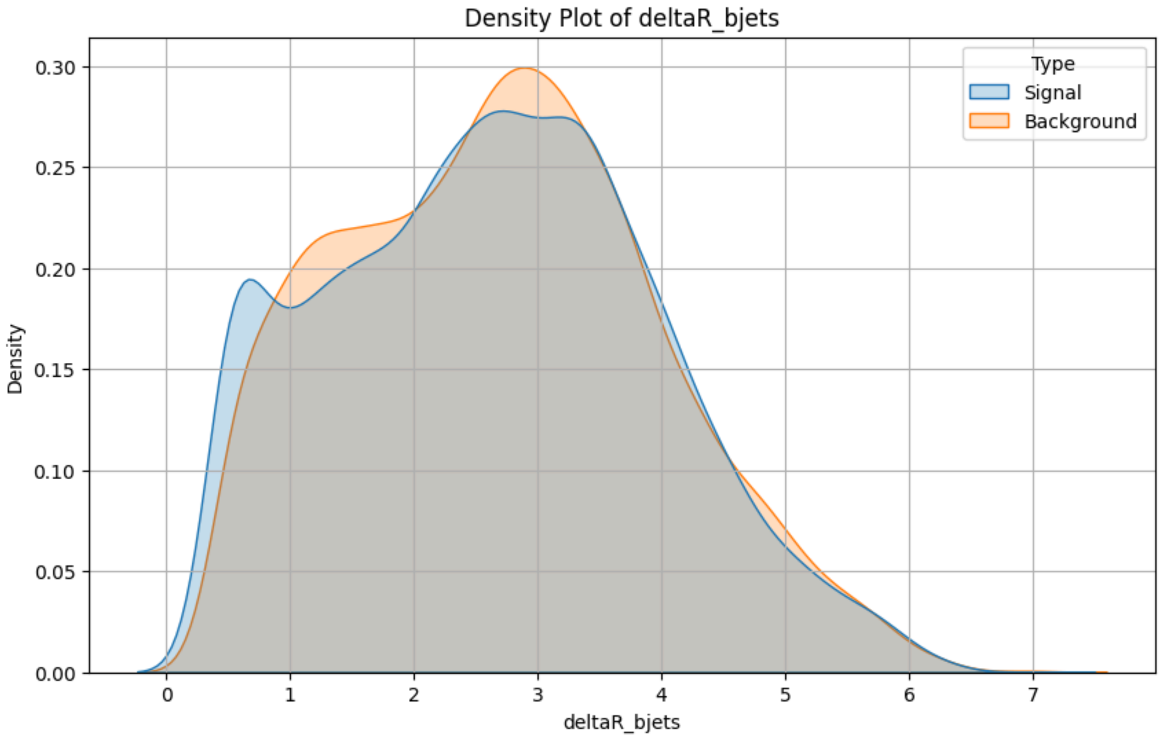


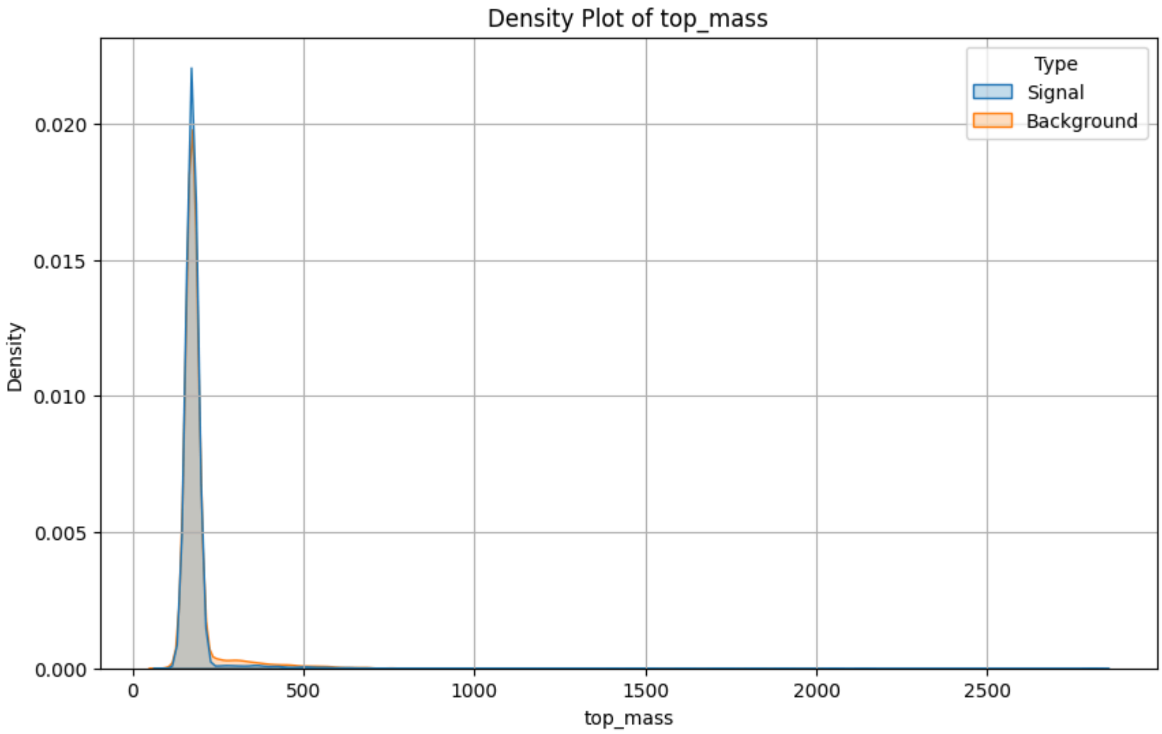
Density Plot of deltaEta_bjets
Type
Signal
Background
Density
deltaEta_bjets
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
−6
−4
−2
0
2
4



Density Plot of deltaPhi_bjets
Type
Signal
Background
Density
deltaPhi_bjets
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
−8
−6
−4
−2
0
2
4
6
8


Density Plot of deltaR_jets
Type
Signal
Background
Density
deltaR_jets
0.0
0.1
0.2
0.3
0.4
0.5
0
1
2
3
4
5
6
7


Density Plot of deltaR_bjets
Type
Signal
Background
Density
deltaR_bjets
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7



	Variable	Signal Mean	Signal Std Dev	Background Mean
0	jet1_pt	327.913806	210.909642	174.845906
1	jet2_pt	227.726548	138.746388	128.249269
2	jet3_pt	160.075479	70.473132	95.247804
3	numJets	11.907100	1.867784	7.992100
4	numBJets	3.220000	1.346618	2.628500
5	bjet1_pt	228.028349	160.211425	141.447562
6	sumbjet_pt	431.375690	293.190688	255.092182
7	Scalar_HT	1307.858939	471.347441	648.850664
8	missing_ET	59.393203	48.249683	37.499789
9	rat_MET_HT	1.635210	1.181753	1.487517
10	deltaEta_jets	0.001854	1.149845	-0.001165
11	deltaPhi_jets	-0.052621	2.938624	0.007301
12	deltaEta_bjets	0.014526	1.178438	-0.008204
13	deltaPhi_bjets	-0.065385	2.699535	0.009292
14	deltaR_jets	2.990983	1.006753	3.015709
15	deltaR_bjets	2.641749	1.304285	2.691509
16	top_mass	188.448369	99.765615	195.010709

	Background Std Dev	Signal IQR	Background IQR
0	100.716293	214.829422	90.726904
1	66.860378	120.172493	63.499533
2	41.381533	72.788029	44.550486
3	1.695501	2.000000	2.000000
4	1.188036	2.000000	1.000000
5	82.622212	145.978680	81.873920
6	165.144226	318.522884	184.454403
7	269.878435	561.555435	297.686256
8	27.502012	47.863606	28.774477
9	0.981800	1.293434	1.151557
10	1.178140	1.462717	1.444782
11	2.952025	5.749130	5.753300
12	1.045917	1.494457	1.374553
13	2.786250	4.685269	4.927325
14	1.003550	1.126661	1.191551
15	1.269779	1.943626	1.877583
16	101.202783	7.724556	11.058369

Random Forest Classifier and Feature Selection

In this section, we will utilize the Random Forest Classifier to analyze our dataset. Random forests are a versatile machine learning method that can be used for both classification and regression tasks. We will also perform feature selection using mutual information, which helps in identifying the most significant features influencing the target variable.

First, we will import the necessary libraries and load our dataset. Then we will split the data into training and testing sets, train the Random Forest model, and evaluate its performance based on accuracy.

In [6]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import mutual_info_classif
```

Feature Selection

In this section, we will perform data analysis on our signal and background datasets by loading them from Excel files, labeling the events, and combining them into a single DataFrame. We will then split the data into features and target variables, followed by training a Random Forest Classifier to distinguish between signal and background events. Additionally, we will extract and display the feature importances to understand the significant factors influencing our model.

In [7]:

```
# Define feature columns (all variables except 'Type')
features = combined_df.columns.drop('Type')

# Calculate mutual information
X = combined_df[features]
y = combined_df['Type']
mi_scores = mutual_info_classif(X, y, random_state=42)

# Create a DataFrame for easier interpretation
mi_scores_df = pd.DataFrame({'Variable': features, 'MI Score': mi_scores}
mi_scores_df = mi_scores_df.sort_values(by='MI Score', ascending=False)

print("Mutual Information Scores:")
print(mi_scores_df)
```

```
Mutual Information Scores:
          Variable  MI Score
20          Weight  0.695897
6          numJets  0.374010
10       Scalar_HT  0.355612
5          jet6_pt  0.303628
4          jet5_pt  0.271671
3          jet4_pt  0.232014
2          jet3_pt  0.202610
1          jet2_pt  0.170525
0          jet1_pt  0.155703
8         bjet1_pt  0.119059
9       sumbjet_pt  0.081282
18    deltaR_bjets  0.055383
15  deltaEta_bjets  0.053307
16  deltaPhi_bjets  0.050827
11      missing_ET  0.044009
7         numBJets  0.027287
19        top_mass  0.019019
17     deltaR_jets  0.005844
14   deltaPhi_jets  0.004335
12      rat_MET_HT  0.000000
13   deltaEta_jets  0.000000
```

In [8]:

```python
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,

# Train a Random Forest Classifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Get feature importance from the trained model
feature_importances = rf.feature_importances_
importance_df = pd.DataFrame({'Variable': features, 'Importance': feature
importance_df = importance_df.sort_values(by='Importance', ascending=Fals

print("Feature Importance from Random Forest:")
print(importance_df)
```

```
Feature Importance from Random Forest:
          Variable  Importance
20          Weight    0.494300
6          numJets    0.164949
10       Scalar_HT    0.101357
4          jet5_pt    0.067901
5          jet6_pt    0.062299
3          jet4_pt    0.028600
1          jet2_pt    0.021195
2          jet3_pt    0.019844
0          jet1_pt    0.012760
8         bjet1_pt    0.004579
9       sumbjet_pt    0.003239
11      missing_ET    0.003188
19        top_mass    0.002772
13   deltaEta_jets    0.001965
15  deltaEta_bjets    0.001808
18    deltaR_bjets    0.001788
14   deltaPhi_jets    0.001649
17     deltaR_jets    0.001628
16  deltaPhi_bjets    0.001599
12      rat_MET_HT    0.001524
7         numBJets    0.001057
```

In [10]:

```
# Select the top variables for further analysis (adjust as needed)
top_variables = mi_scores_df.head(5)['Variable'].tolist()
print("Top Variables for Further Analysis:", top_variables)
```

```
Top Variables for Further Analysis: ['Weight', 'numJets', 'Scalar_HT', 'je
t6_pt', 'jet5_pt']
```

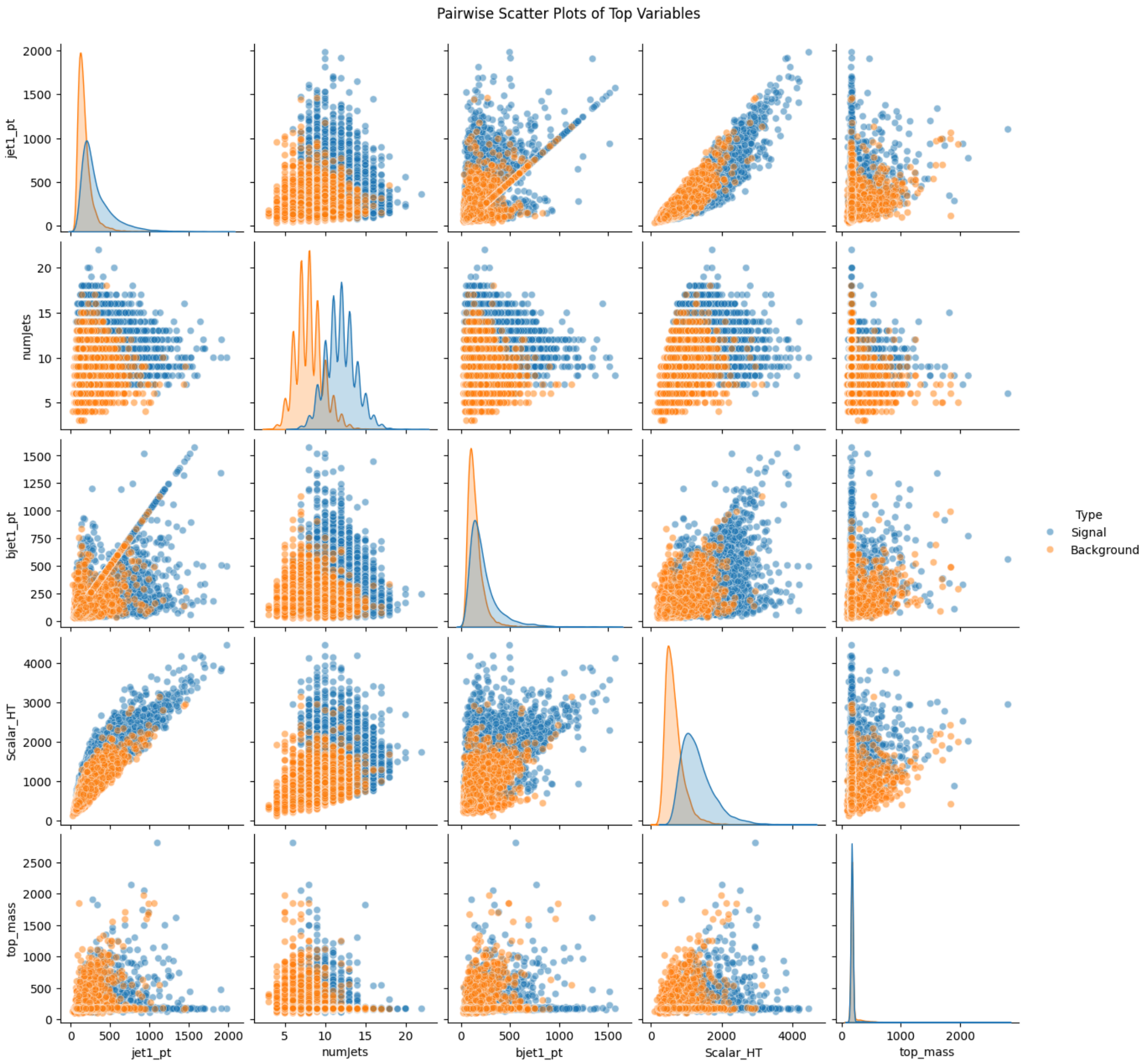
Pairwise Scatter Plots and Correlation Heatmap

In this section, we will visualize the relationships between the top variables in our dataset. We will create pairwise scatter plots to observe how these variables interact with each other, distinguishing between signal and background events using color coding. Additionally, we will generate a correlation heatmap to examine the correlation coefficients among the selected top variables and the target variable. This will help us understand which variables are strongly correlated and may influence the classification model.

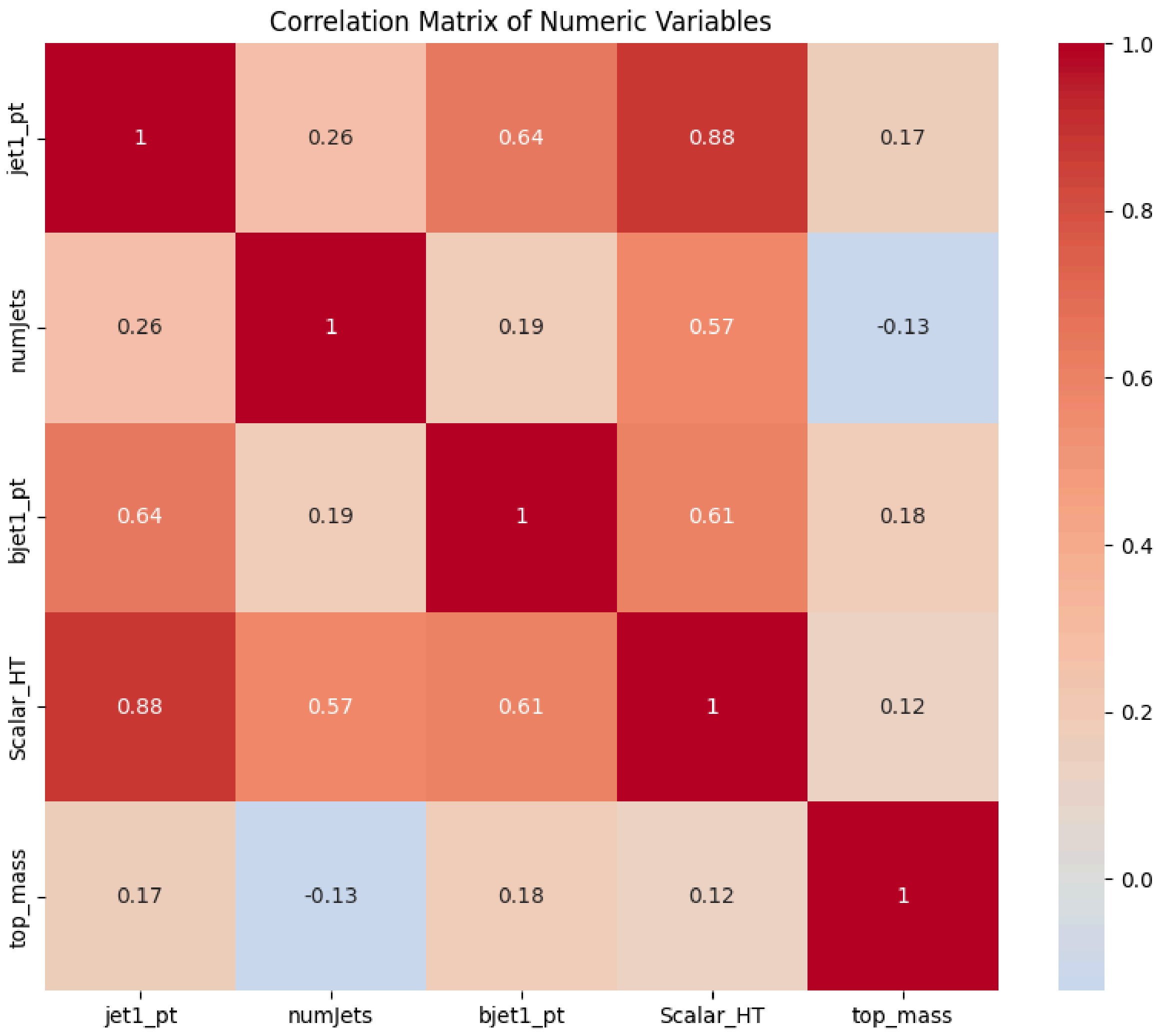
In [11]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# Load your data (if not already done)
signal_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/SignalTTr
background_df = pd.read_excel("/home/syed/Downloads/Dataset/weights/HBack


# Combine the data into a single DataFrame
signal_df['Type'] = 'Signal'
background_df['Type'] = 'Background'
combined_df = pd.concat([signal_df, background_df], ignore_index=True)

# Specify the top variables based on previous analysis
```

```
top_variables = ['jet1_pt', 'numJets', 'bjet1_pt', 'Scalar_HT', 'top_mass

# Create pairwise scatter plots for the top variables
sns.pairplot(combined_df, vars=top_variables, hue='Type', plot_kws={'alph
plt.suptitle('Pairwise Scatter Plots of Top Variables', y=1.02)
plt.show()

# Summary of pairwise plots
print("### Pairwise Scatter Plots Summary ###")
print("1. **jet1_pt vs numJets**: This plot shows a positive correlation
print("2. **bjet1_pt vs Scalar_HT**: There is a clear separation between
print("3. **top_mass**: The distribution of top mass shows some overlap b

# Create a correlation heatmap for the top variables
plt.figure(figsize=(10, 8))

# Calculate the correlation matrix
corr_matrix = combined_df[top_variables].corr()

# Create heatmap
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Matrix of Numeric Variables')
plt.show()

# Summary of correlation matrix
print("### Correlation Matrix Summary ###")
for i in range(len(top_variables)):
    for j in range(i+1, len(top_variables)):
        corr_value = corr_matrix.iloc[i, j]
        if abs(corr_value) > 0.5:
            relationship = "strong positive" if corr_value > 0 else "stro
            print(f"**{top_variables[i]} vs {top_variables[j]}**: There i
print("Variables with a correlation close to 0 indicate little to no line
```




```
### Pairwise Scatter Plots Summary ###
1. **jet1_pt vs numJets**: This plot shows a positive correlation between jet1 momentum and the number of jets. Signal events have higher values for both variables.
2. **bjet1_pt vs Scalar_HT**: There is a clear separation between signal and background events, with signal events generally exhibiting higher b-jet momentum and Scalar_HT values.
3. **top_mass**: The distribution of top mass shows some overlap between signal and background, indicating that this variable may not be as effective in separating the two event types.
```




```
### Correlation Matrix Summary ###
**jet1_pt vs bjet1_pt**: There is a strong positive correlation (r = 0.6
4).
**jet1_pt vs Scalar_HT**: There is a strong positive correlation (r = 0.8
8).
**numJets vs Scalar_HT**: There is a strong positive correlation (r = 0.5
7).
**bjet1_pt vs Scalar_HT**: There is a strong positive correlation (r = 0.6
1).
Variables with a correlation close to 0 indicate little to no linear relat
ionship.
```