itsm-ai-enhancement-initiative-1

September 26, 2024

[61]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows',500)
```

[62]:
```
data = pd.read_csv('my_data2.csv')
```

[63]:
```
data
```

[63]:

	CI_Name	CI_Cat	CI_Subcat	WBS	\
0	SUB000508	subapplication	Web Based Application	WBS000162	
1	WBA000124	application	Web Based Application	WBS000088	
2	DTA000024	application	Desktop Application	WBS000092	
3	WBA000124	application	Web Based Application	WBS000088	
4	WBA000124	application	Web Based Application	WBS000088	
…	…	…	…	…	
46601	SBA000464	application	Server Based Application	WBS000073	
46602	SBA000461	application	Server Based Application	WBS000073	
46603	LAP000019	computer	Laptop	WBS000091	
46604	WBA000058	application	Web Based Application	WBS000073	
46605	DCE000077	hardware	DataCenterEquipment	WBS000267	

	Incident_ID	Status	Impact	Urgency	Priority	number_cnt	\
0	IM0000004	Closed	4	4	4.0	0.601292	
1	IM0000005	Closed	3	3	3.0	0.415050	
2	IM0000006	Closed	NS	3	NaN	0.517551	
3	IM0000011	Closed	4	4	4.0	0.642927	
4	IM0000012	Closed	4	4	4.0	0.345258	
…	…	…	…	…	…	…	
46601	IM0047053	Closed	4	4	4.0	0.231896	
46602	IM0047054	Closed	4	4	4.0	0.805153	
46603	IM0047055	Closed	5	5	5.0	0.917466	
46604	IM0047056	Closed	4	4	4.0	0.701278	

```
46605   IM0047057  Closed      3       3       3.0    0.902320

                     Category  KB_number Alert_Status  No_of_Reassignments  \
0                    incident  KM0000553       closed                 26.0
1                    incident  KM0000611       closed                 33.0
2      request for information  KM0000339       closed                  3.0
3                    incident  KM0000611       closed                 13.0
4                    incident  KM0000611       closed                  2.0
…                       …        …          …                    …
46601                incident  KM0001314       closed                  0.0
46602                incident  KM0002360       closed                  0.0
46603                incident  KM0000315       closed                  0.0
46604                incident  KM0001287       closed                  0.0
46605                incident  KM0000182       closed                  0.0

              Open_Time       Reopen_Time     Resolved_Time        Close_Time  \
0      05-02-2012 13:32               NaN  04-11-2013 13:50  04-11-2013 13:51
1      12-03-2012 15:44  02-12-2013 12:31  02-12-2013 12:36  02-12-2013 12:36
2      29-03-2012 12:36               NaN  13-01-2014 15:12  13-01-2014 15:13
3      17-07-2012 11:49               NaN  14-11-2013 09:31  14-11-2013 09:31
4      10-08-2012 11:01               NaN  08-11-2013 13:55  08-11-2013 13:55
…                  …                 …                …                 …
46601  31-03-2014 16:23               NaN  31-03-2014 16:29  31-03-2014 16:29
46602  31-03-2014 15:03               NaN  31-03-2014 15:29  31-03-2014 15:29
46603  31-03-2014 15:28               NaN  31-03-2014 15:32  31-03-2014 15:32
46604  31-03-2014 15:35               NaN  31-03-2014 15:42  31-03-2014 15:42
46605  31-03-2014 17:24               NaN  31-03-2014 22:47  31-03-2014 22:47

      Handle_Time_hrs                  Closure_Code  \
0      3,87,16,91,111                         Other
1      4,35,47,86,389                      Software
2      4,84,31,19,444  No error - works as designed
3      4,32,18,33,333                Operator error
4      3,38,39,03,333                         Other
…                 …                             …
46601          0,095                         Other
46602     0,428333333                    User error
46603     0,071666667                      Hardware
46604     0,116944444                      Software
46605     0,586388889                      Hardware

       No_of_Related_Interactions Related_Interaction  \
0                             1.0           SD0000007
1                             1.0           SD0000011
2                             1.0           SD0000017
3                             1.0           SD0000025
4                             1.0           SD0000029
```

```
…                              …                 …
46601                          1.0           SD0147021
46602                          1.0           SD0146967
46603                          1.0           SD0146982
46604                          1.0           SD0146986
46605                          1.0           SD0147088

       No_of_Related_Incidents  No_of_Related_Changes Related_Change
0                          2.0                   NaN            NaN
1                          1.0                   NaN            NaN
2                          NaN                   NaN            NaN
3                          NaN                   NaN            NaN
4                          NaN                   NaN            NaN
…                            …                     …              …
46601                      NaN                   NaN            NaN
46602                      NaN                   NaN            NaN
46603                      NaN                   NaN            NaN
46604                      NaN                   NaN            NaN
46605                      NaN                   NaN            NaN

[46606 rows x 25 columns]
```

[64]:
```
for column in data.columns:
    unique_value = data[column].unique()
    print(f"Unique values in column '{column}':")
    print(unique_value)
    print(f"Total Unique values in column '{column}':" , len(unique_value))
    print("\n")
```

```
Unique values in column 'CI_Name':
['SUB000508' 'WBA000124' 'DTA000024' … 'CBD000595' 'CBD000443'
 'SWT000008']
Total Unique values in column 'CI_Name': 3019


Unique values in column 'CI_Cat':
['subapplication' 'application' 'computer' nan 'displaydevice' 'software'
 'storage' 'database' 'hardware' 'officeelectronics' 'networkcomponents'
 'applicationcomponent' 'Phone']
Total Unique values in column 'CI_Cat': 13


Unique values in column 'CI_Subcat':
['Web Based Application' 'Desktop Application' 'Server Based Application'
 'SAP' 'Client Based Application' 'Citrix' 'Standard Application'
 'Windows Server' 'Laptop' 'Linux Server' nan 'Monitor'
 'Automation Software' 'SAN' 'Banking Device' 'Desktop' 'Database'
```

```
 'Oracle Server' 'Keyboard' 'Printer' 'Exchange' 'System Software' 'VDI'
 'Encryption' 'Omgeving' 'MigratieDummy' 'Scanner' 'Controller'
 'DataCenterEquipment' 'KVM Switches' 'Switch' 'Database Software'
 'Network Component' 'Unix Server' 'Lines' 'ESX Cluster' 'zOS Server'
 'SharePoint Farm' 'NonStop Server' 'Application Server'
 'Security Software' 'Thin Client' 'zOS Cluster' 'Router' 'VMWare'
 'Net Device' 'Neoview Server' 'MQ Queue Manager' 'UPS' 'Number'
 'Iptelephony' 'Windows Server in extern beheer' 'Modem' 'X86 Server'
 'ESX Server' 'Virtual Tape Server' 'IPtelephony' 'NonStop Harddisk'
 'Firewall' 'RAC Service' 'zOS Systeem' 'Instance' 'NonStop Storage'
 'Protocol' 'Tape Library']
Total Unique values in column 'CI_Subcat': 65


Unique values in column 'WBS':
['WBS000162' 'WBS000088' 'WBS000092' 'WBS000055' 'WBS000090' 'WBS000073'
 'WBS000066' 'WBS000071' 'WBS000263' 'WBS000072' 'WBS000054' 'WBS000271'
 'WBS000170' 'WBS000014' 'WBS000098' 'WBS000043' 'WBS000086' 'WBS000027'
 'WBS000300' 'WBS000223' 'WBS000311' 'WBS000142' 'WBS000109' 'WBS000153'
 'WBS000102' 'WBS000152' 'WBS000129' 'WBS000251' 'WBS000115' 'WBS000168'
 'WBS000136' 'WBS000314' 'WBS000187' 'WBS000135' 'WBS000217' 'WBS000091'
 'WBS000296' 'WBS000161' 'WBS000312' 'WBS000012' 'WBS000234' 'WBS000094'
 'WBS000318' 'WBS000285' 'WBS000093' 'WBS000016' 'WBS000118' 'WBS000298'
 'WBS000096' 'WBS000025' 'WBS000272' 'WBS000284' 'WBS000128' 'WBS000067'
 'WBS000110' 'WBS000292' 'WBS000165' 'WBS000015' 'WBS000111' 'WBS000199'
 'WBS000095' 'WBS000089' 'WBS000017' 'WBS000069' 'WBS000241' 'WBS000023'
 'WBS000140' 'WBS000070' 'WBS000167' 'WBS000138' 'WBS000125' 'WBS000249'
 'WBS000039' 'WBS000048' 'WBS000253' 'WBS000216' 'WBS000186' 'WBS000172'
 'WBS000228' 'WBS000309' 'WBS000132' 'WBS000077' 'WBS000326' 'WBS000099'
 'WBS000224' 'WBS000018' 'WBS000008' 'WBS000030' 'WBS000121' 'WBS000052'
 'WBS000058' 'WBS000256' 'WBS000242' 'WBS000123' 'WBS000047' 'WBS000146'
 'WBS000307' 'WBS000075' 'WBS000013' 'WBS000063' 'WBS000239' 'WBS000218'
 'WBS000148' 'WBS000083' 'WBS000139' 'WBS000258' 'WBS000053' 'WBS000006'
 'WBS000169' 'WBS000182' 'WBS000255' 'WBS000268' 'WBS000267' 'WBS000294'
 'WBS000273' 'WBS000037' 'WBS000062' 'WBS000151' 'WBS000133' 'WBS000171'
 'WBS000163' 'WBS000212' 'WBS000174' 'WBS000245' 'WBS000265' 'WBS000007'
 'WBS000130' 'WBS000134' 'WBS000331' 'WBS000264' 'WBS000281' 'WBS000042'
 'WBS000081' 'WBS000149' 'WBS000244' 'WBS000044' 'WBS000177' 'WBS000166'
 'WBS000040' 'WBS000201' 'WBS000076' 'WBS000101' 'WBS000183' 'WBS000299'
 'WBS000275' 'WBS000222' 'WBS000022' 'WBS000185' 'WBS000219' 'WBS000237'
 'WBS000316' 'WBS000141' 'WBS000295' 'WBS000100' 'WBS000158' 'WBS000050'
 'WBS000026' 'WBS000280' 'WBS000035' 'WBS000203' 'WBS000181' 'WBS000236'
 'WBS000108' 'WBS000191' 'WBS000033' 'WBS000056' 'WBS000206' 'WBS000304'
 'WBS000279' 'WBS000127' 'WBS000019' 'WBS000270' 'WBS000029' 'WBS000147'
 'WBS000145' 'WBS000005' 'WBS000215' 'WBS000207' 'WBS000204' 'WBS000247'
 'WBS000126' 'WBS000011' 'WBS000107' 'WBS000305' 'WBS000257' 'WBS000059'
 'WBS000306' 'WBS000321' 'WBS000157' 'WBS000079' 'WBS000085' 'WBS000097'
 'WBS000252' 'WBS000061' 'WBS000002' 'WBS000197' 'WBS000250' 'WBS000303'
```

```
 'WBS000188' 'WBS000289' 'WBS000184' 'WBS000301' 'WBS000278' 'WBS000176'
 'WBS000120' 'WBS000137' 'WBS000113' 'WBS000122' 'WBS000328' 'WBS000112'
 'WBS000192' 'WBS000221' 'WBS000209' 'WBS000330' 'WBS000057' 'WBS000060'
 'WBS000310' 'WBS000131' 'WBS000175' 'WBS000160' 'WBS000319' 'WBS000104'
 'WBS000010' 'WBS000211' 'WBS000087' 'WBS000277' 'WBS000243' 'WBS000196'
 'WBS000200' 'WBS000031' 'WBS000315' 'WBS000214' 'WBS000119' 'WBS000254'
 'WBS000240' 'WBS000117' 'WBS000021' 'WBS000068' 'WBS000220' 'WBS000106'
 'WBS000282' 'WBS000194' 'WBS000232' 'WBS000246' 'WBS000020' 'WBS000150'
 'WBS000001' 'WBS000325' 'WBS000235' 'WBS000195' 'WBS000302' 'WBS000322'
 'WBS000248' 'WBS000004' 'WBS000329' 'WBS000144' 'WBS000051' 'WBS000074'
 'WBS000317' 'WBS000286' 'WBS000339' 'WBS000335' 'WBS000024' 'WBS000337'
 'WBS000156' 'WBS000164' 'WBS000210' 'WBS000320' 'WBS000114' 'WBS000262'
 'WBS000327' 'WBS000332' 'WBS000313' 'WBS000105']
Total Unique values in column 'WBS': 274


Unique values in column 'Incident_ID':
['IM0000004' 'IM0000005' 'IM0000006' … 'IM0047055' 'IM0047056'
 'IM0047057']
Total Unique values in column 'Incident_ID': 46606


Unique values in column 'Status':
['Closed' 'Work in progress']
Total Unique values in column 'Status': 2


Unique values in column 'Impact':
['4' '3' 'NS' '5' '2' '1']
Total Unique values in column 'Impact': 6


Unique values in column 'Urgency':
[4 3 5 2 1 '5' '3' '4' '2' '1' '5 - Very Low']
Total Unique values in column 'Urgency': 11


Unique values in column 'Priority':
[ 4.  3. nan  5.  2.  1.]
Total Unique values in column 'Priority': 6


Unique values in column 'number_cnt':
[0.60129228 0.41504997 0.51755133 … 0.91746629 0.70127816 0.90231951]
Total Unique values in column 'number_cnt': 46606


Unique values in column 'Category':
```

```
['incident' 'request for information' 'complaint' 'request for change']
Total Unique values in column 'Category': 4


Unique values in column 'KB_number':
['KM0000553' 'KM0000611' 'KM0000339' … 'KM0001052' 'KM0000378'
 'KM0001578']
Total Unique values in column 'KB_number': 1825


Unique values in column 'Alert_Status':
['closed']
Total Unique values in column 'Alert_Status': 1


Unique values in column 'No_of_Reassignments':
[26. 33.  3. 13.  2.  4.  5.  6.  8. 17.  1.  7. 12.  0. 11.  9. 25. 30.
 15. 37. 32. 22. 10. 21. 19. 14. 46. 18. 16. 42. 23. 39. 20. 45. 38. 24.
 34. 29. 27. 31. nan 36.]
Total Unique values in column 'No_of_Reassignments': 42


Unique values in column 'Open_Time':
['05-02-2012 13:32' '12-03-2012 15:44' '29-03-2012 12:36' …
 '31-03-2014 15:28' '31-03-2014 15:35' '31-03-2014 17:24']
Total Unique values in column 'Open_Time': 34636


Unique values in column 'Reopen_Time':
[nan '02-12-2013 12:31' '28-01-2014 14:07' … '31-03-2014 10:20'
 '31-03-2014 10:55' '31-03-2014 16:21']
Total Unique values in column 'Reopen_Time': 2245


Unique values in column 'Resolved_Time':
['04-11-2013 13:50' '02-12-2013 12:36' '13-01-2014 15:12' …
 '31-03-2014 15:29' '31-03-2014 15:42' '31-03-2014 22:47']
Total Unique values in column 'Resolved_Time': 33628


Unique values in column 'Close_Time':
['04-11-2013 13:51' '02-12-2013 12:36' '13-01-2014 15:13' …
 '31-03-2014 16:29' '31-03-2014 15:32' '31-03-2014 22:47']
Total Unique values in column 'Close_Time': 34528


Unique values in column 'Handle_Time_hrs':
['3,87,16,91,111' '4,35,47,86,389' '4,84,31,19,444' … '1,10,13,88,889'
```

```
 '1,99,30,55,556' '0,428333333']
Total Unique values in column 'Handle_Time_hrs': 30639


Unique values in column 'Closure_Code':
['Other' 'Software' 'No error - works as designed' 'Operator error'
 'Unknown' 'Data' 'Referred' 'Hardware' 'Questions' 'User error' 'Inquiry'
 'User manual not used' 'Kwaliteit van de output' nan 'Overig']
Total Unique values in column 'Closure_Code': 15


Unique values in column 'No_of_Related_Interactions':
[  1.   2.   3.  14.   7.   4.   5. 370.   9.  11.  54.  nan 288.  34.
  44.  39.  12.  42.   6.   8.  28.  29.  13.  18.  20.  15.  31.  88.
  30.  41.  74.  16.  24.  17.  57.  33.  10. 118.  45.  55.  40.  19.
  22.  23.  37.  26.  43.  25.  27.  21.]
Total Unique values in column 'No_of_Related_Interactions': 50


Unique values in column 'Related_Interaction':
['SD0000007' 'SD0000011' 'SD0000017' … 'SD0146982' 'SD0146986'
 'SD0147088']
Total Unique values in column 'Related_Interaction': 43060


Unique values in column 'No_of_Related_Incidents':
[ 2.  1. nan 23.  4.  3.  7. 11. 12. 54. 24. 16.  8.  6. 25. 14. 10.  9.
  5. 17. 26. 21. 63. 13. 15.]
Total Unique values in column 'No_of_Related_Incidents': 25


Unique values in column 'No_of_Related_Changes':
[nan  1.  2.  3.  9.]
Total Unique values in column 'No_of_Related_Changes': 5


Unique values in column 'Related_Change':
[nan 'C00000056' '#MULTIVALUE' 'C00000308' 'C00000582' 'C00006478'
 'C00000778' 'C00002486' 'C00000577' 'C00001219' 'C00004614' 'C00000510'
 'C00002078' 'C00001452' 'C00000589' 'C00004490' 'C00002426' 'C00000517'
 'C00000449' 'C00001728' 'C00001549' 'C00001953' 'C00002007' 'C00002268'
 'C00001250' 'C00003624' 'C00001667' 'C00002178' 'C00001730' 'C00000527'
 'C00002657' 'C00001026' 'C00014458' 'C00001135' 'C00004384' 'C00003123'
 'C00000713' 'C00003040' 'C00002301' 'C00000596' 'C00004344' 'C00001062'
 'C00006745' 'C00003468' 'C00003404' 'C00002804' 'C00005369' 'C00003547'
 'C00001137' 'C00004090' 'C00002337' 'C00001455' 'C00007202' 'C00000874'
 'C00001215' 'C00005866' 'C00004950' 'C00002375' 'C00001807' 'C00002389'
 'C00004493' 'C00001507' 'C00004739' 'C00004854' 'C00004994' 'C00005110'
```

```
 'C00003013' 'C00005050' 'C00001831' 'C00005461' 'C00004549' 'C00000360'
 'C00006422' 'C00007055' 'C00005847' 'C00004385' 'C00005815' 'C00004294'
 'C00006302' 'C00011501' 'C00006824' 'C00000633' 'C00001012' 'C00000816'
 'C00006883' 'C00007092' 'C00000628' 'C00005713' 'C00000829' 'C00000122'
 'C00000050' 'C00006599' 'C00007015' 'C00007915' 'C00009722' 'C00008222'
 'C00006823' 'C00007099' 'C00004679' 'C00006401' 'C00000600' 'C00007132'
 'C00002378' 'C00008054' 'C00007263' 'C00007098' 'C00008442' 'C00008486'
 'C00008638' 'C00007161' 'C00008356' 'C00007983' 'C00008467' 'C00007747'
 'C00006448' 'C00008726' 'C00008750' 'C00009025' 'C00009069' 'C00005456'
 'C00008700' 'C00008748' 'C00006833' 'C00013454' 'C00007572' 'C00008787'
 'C00009165' 'C00009563' 'C00009692' 'C00009397' 'C00009567' 'C00009444'
 'C00010116' 'C00009448' 'C00012545' 'C00009656' 'C00009761' 'C00009712'
 'C00009821' 'C00010379' 'C00015705' 'C00009947' 'C00005261' 'C00010259'
 'C00010785' 'C00010344' 'C00010182' 'C00003702' 'C00010314' 'C00010749'
 'C00010740' 'C00010081' 'C00015140' 'C00011366' 'C00014075' 'C00011182'
 'C00013072' 'C00011406' 'C00011591' 'C00010941' 'C00012038' 'C00012048'
 'C00009966' 'C00012194' 'C00013273' 'C00012062' 'C00012116' 'C00012730'
 'C00011170' 'C00014035' 'C00011858' 'C00013064' 'C00013104' 'C00012714'
 'C00004044' 'C00013379' 'C00013867' 'C00013595' 'C00016295' 'C00013740'
 'C00013125' 'C00013606' 'C00013982' 'C00015047' 'C00014624' 'C00014122'
 'C00014301' 'C00014221' 'C00014707' 'C00014360' 'C00014375' 'C00014475'
 'C00014622' 'C00015040' 'C00014876' 'C00014661' 'C00014762' 'C00015613'
 'C00014296' 'C00014981' 'C00015091' 'C00005858' 'C00012547' 'C00015923'
 'C00015776' 'C00015609' 'C00018435' 'C00015758' 'C00016192' 'C00015759'
 'C00017230' 'C00016153' 'C00016233' 'C00015544' 'C00015025' 'C00016781'
 'C00016571' 'C00012923' 'C00016689' 'C00017136' 'C00017161' 'C00017553'
 'C00018471' 'C00017031' 'C00017302' 'C00018294' 'C00018267' 'C00018421'
 'C00018403' 'C00018549' 'C00017321' 'C00017594' 'C00000385']
Total Unique values in column 'Related_Change': 233
```

[65]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46606 entries, 0 to 46605
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   CI_Name                     46606 non-null  object
 1   CI_Cat                      46495 non-null  object
 2   CI_Subcat                   46495 non-null  object
 3   WBS                         46606 non-null  object
 4   Incident_ID                 46606 non-null  object
 5   Status                      46606 non-null  object
 6   Impact                      46606 non-null  object
 7   Urgency                     46606 non-null  object
 8   Priority                    45226 non-null  float64
```

```
 9   number_cnt                  46606 non-null  float64
 10  Category                    46606 non-null  object
 11  KB_number                   46606 non-null  object
 12  Alert_Status                46606 non-null  object
 13  No_of_Reassignments         46605 non-null  float64
 14  Open_Time                   46606 non-null  object
 15  Reopen_Time                 2284 non-null   object
 16  Resolved_Time               44826 non-null  object
 17  Close_Time                  46606 non-null  object
 18  Handle_Time_hrs             46605 non-null  object
 19  Closure_Code                46146 non-null  object
 20  No_of_Related_Interactions  46492 non-null  float64
 21  Related_Interaction         46606 non-null  object
 22  No_of_Related_Incidents     1222 non-null   float64
 23  No_of_Related_Changes       560 non-null    float64
 24  Related_Change              560 non-null    object
dtypes: float64(6), object(19)
memory usage: 8.9+ MB
```

[66]:
```
data.isnull().sum()
```

[66]:
```
CI_Name                         0
CI_Cat                        111
CI_Subcat                     111
WBS                             0
Incident_ID                     0
Status                          0
Impact                          0
Urgency                         0
Priority                     1380
number_cnt                      0
Category                        0
KB_number                       0
Alert_Status                    0
No_of_Reassignments             1
Open_Time                       0
Reopen_Time                 44322
Resolved_Time                1780
Close_Time                      0
Handle_Time_hrs                 1
Closure_Code                  460
No_of_Related_Interactions    114
Related_Interaction             0
No_of_Related_Incidents     45384
No_of_Related_Changes       46046
Related_Change              46046
dtype: int64
```

[67]:
```
data.describe()
```

[67]:

	Priority	number_cnt	No_of_Reassignments
count	45226.000000	46606.000000	46605.000000
mean	4.215805	0.499658	1.131831
std	0.705624	0.288634	2.269774
min	1.000000	0.000023	0.000000
25%	4.000000	0.248213	0.000000
50%	4.000000	0.500269	0.000000
75%	5.000000	0.749094	2.000000
max	5.000000	0.999997	46.000000

	No_of_Related_Interactions	No_of_Related_Incidents
count	46492.000000	1222.000000
mean	1.149897	1.669394
std	2.556338	3.339687
min	1.000000	1.000000
25%	1.000000	1.000000
50%	1.000000	1.000000
75%	1.000000	1.000000
max	370.000000	63.000000

	No_of_Related_Changes
count	560.000000
mean	1.058929
std	0.403596
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	9.000000

[68]:
```
# Droping the column
data.drop(columns=['Reopen_Time', 'No_of_Related_Incidents',␣
 ↪'No_of_Related_Changes', 'Related_Change'], inplace=True)
```

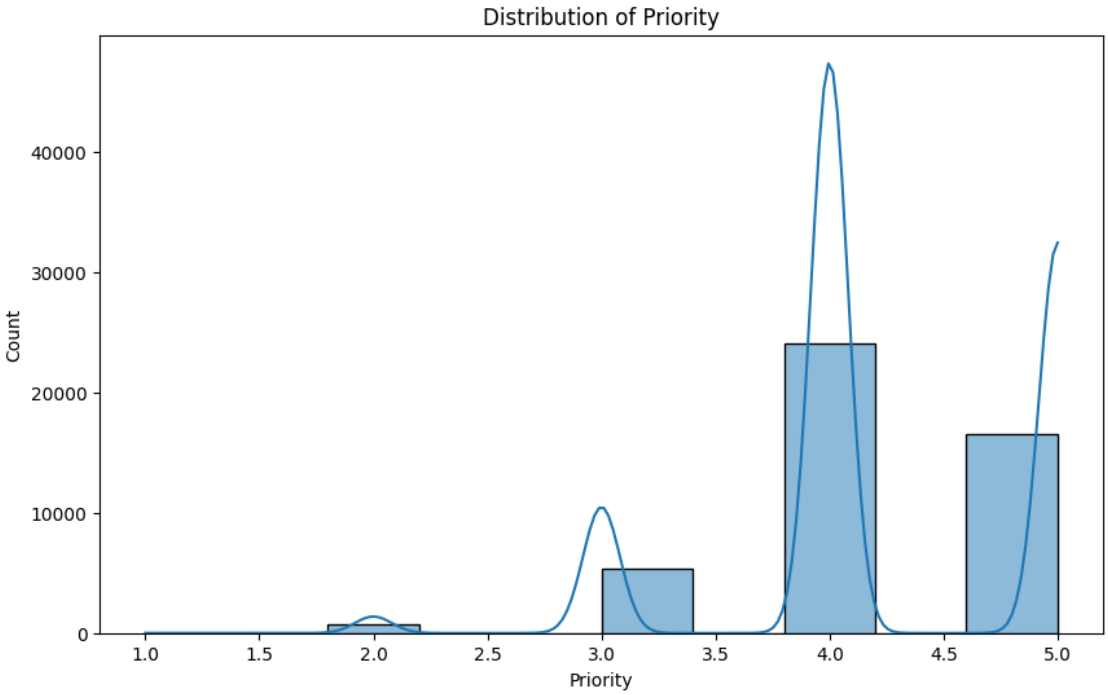
[69]:
```
# Basic cleaning
data['CI_Cat'].fillna('Unknown', inplace=True)
data['CI_Subcat'].fillna('Unknown', inplace=True)

data['Priority'].fillna(data['Priority'].median(), inplace=True)
data['No_of_Reassignments'].fillna(0, inplace=True)

data['Open_Time'] = pd.to_datetime(data['Open_Time'], errors='coerce')
data['Resolved_Time'] = pd.to_datetime(data['Resolved_Time'], errors='coerce')
data['Close_Time'] = pd.to_datetime(data['Close_Time'], errors='coerce')
```

```
data['Resolved_Time'].fillna(data['Close_Time'], inplace=True)
data['Handle_Time_hrs'] = pd.to_numeric(data['Handle_Time_hrs'],␣
 ↪errors='coerce')
data['Handle_Time_hrs'].fillna(data['Handle_Time_hrs'].mean(), inplace=True)
data['Closure_Code'].fillna('Unknown', inplace=True)
data['No_of_Related_Interactions'].fillna(0, inplace=True)
```

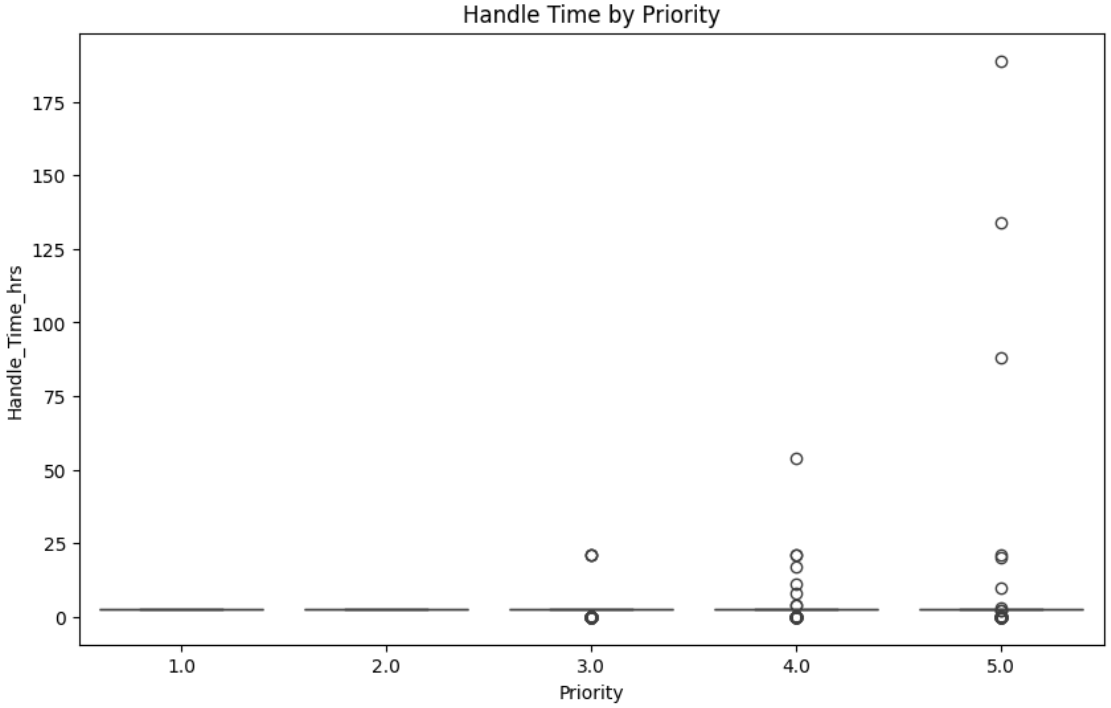
[70]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46606 entries, 0 to 46605
Data columns (total 21 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   CI_Name                      46606 non-null  object
 1   CI_Cat                       46606 non-null  object
 2   CI_Subcat                    46606 non-null  object
 3   WBS                          46606 non-null  object
 4   Incident_ID                  46606 non-null  object
 5   Status                       46606 non-null  object
 6   Impact                       46606 non-null  object
 7   Urgency                      46606 non-null  object
 8   Priority                     46606 non-null  float64
 9   number_cnt                   46606 non-null  float64
 10  Category                     46606 non-null  object
 11  KB_number                    46606 non-null  object
 12  Alert_Status                 46606 non-null  object
 13  No_of_Reassignments          46606 non-null  float64
 14  Open_Time                    18612 non-null  datetime64[ns]
 15  Resolved_Time                18333 non-null  datetime64[ns]
 16  Close_Time                   18333 non-null  datetime64[ns]
 17  Handle_Time_hrs              46606 non-null  float64
 18  Closure_Code                 46606 non-null  object
 19  No_of_Related_Interactions   46606 non-null  float64
 20  Related_Interaction          46606 non-null  object
dtypes: datetime64[ns](3), float64(5), object(13)
memory usage: 7.5+ MB
```

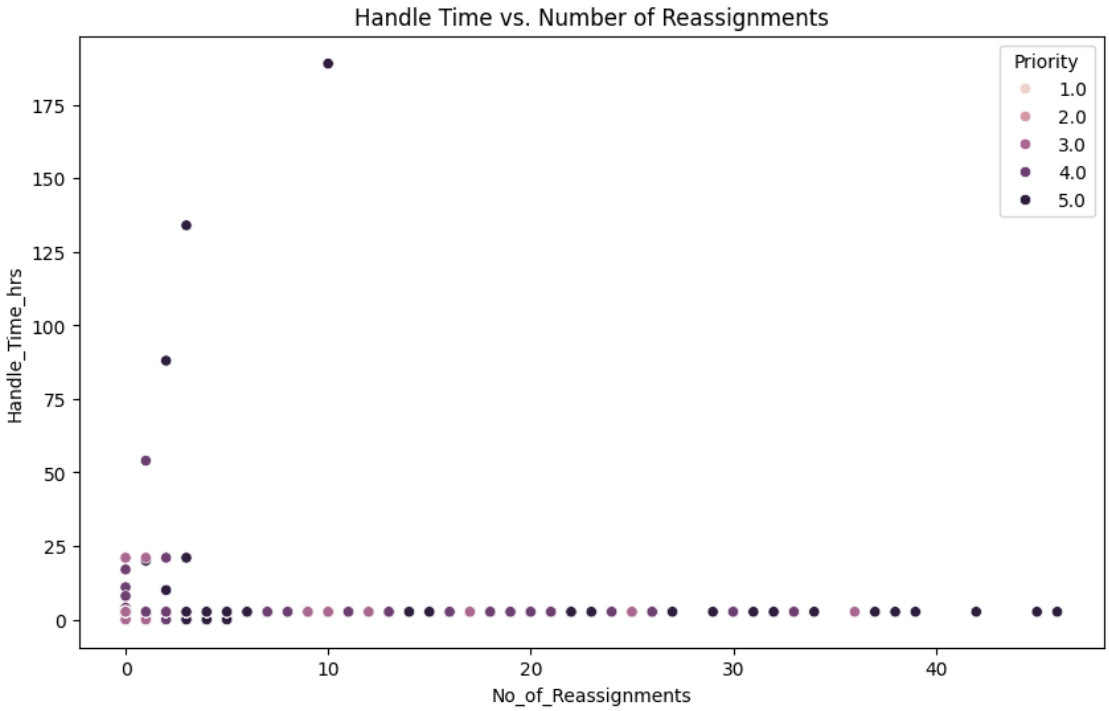
[71]:
```
# Visualizing distributions
plt.figure(figsize=(10, 6))
sns.histplot(data['Priority'], bins=10, kde=True)
plt.title('Distribution of Priority')
plt.show()
```



[72]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='Priority', y='Handle_Time_hrs', data=data)
plt.title('Handle Time by Priority')
plt.show()
```



[73]:
```
# Scatter plot to identify patterns or outliers
plt.figure(figsize=(10, 6))
sns.scatterplot(x='No_of_Reassignments', y='Handle_Time_hrs', hue='Priority',␣
 ↪data=data)
plt.title('Handle Time vs. Number of Reassignments')
plt.show()
```



[74]:
```
def handle_outliers(data, column, method='IQR', factor=1.5):
    if method == 'IQR':
        Q1 = data[column].quantile(0.25)
        Q3 = data[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - factor * IQR
        upper_bound = Q3 + factor * IQR
    elif method == 'z-score':
        mean = data[column].mean()
        std = data[column].std()
        lower_bound = mean - factor * std
        upper_bound = mean + factor * std
    else:
        raise ValueError("Method should be either 'IQR' or 'z-score'")
```

```
    data[column] = np.where(data[column] < lower_bound, lower_bound,␣
 ↪data[column])
    data[column] = np.where(data[column] > upper_bound, upper_bound,␣
 ↪data[column])
    return data
```

[75]:
```
# Handling outliers in numerical columns
numerical_columns = ['Priority', 'No_of_Reassignments', 'Handle_Time_hrs',␣
 ↪'No_of_Related_Interactions']
for column in numerical_columns:
    data = handle_outliers(data, column, method='IQR', factor=1.5)
```

[76]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46606 entries, 0 to 46605
Data columns (total 21 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   CI_Name                     46606 non-null  object
 1   CI_Cat                      46606 non-null  object
 2   CI_Subcat                   46606 non-null  object
 3   WBS                         46606 non-null  object
 4   Incident_ID                 46606 non-null  object
 5   Status                      46606 non-null  object
 6   Impact                      46606 non-null  object
 7   Urgency                     46606 non-null  object
 8   Priority                    46606 non-null  float64
 9   number_cnt                  46606 non-null  float64
 10  Category                    46606 non-null  object
 11  KB_number                   46606 non-null  object
 12  Alert_Status                46606 non-null  object
 13  No_of_Reassignments         46606 non-null  float64
 14  Open_Time                   18612 non-null  datetime64[ns]
 15  Resolved_Time               18333 non-null  datetime64[ns]
 16  Close_Time                  18333 non-null  datetime64[ns]
 17  Handle_Time_hrs             46606 non-null  float64
 18  Closure_Code                46606 non-null  object
 19  No_of_Related_Interactions  46606 non-null  float64
 20  Related_Interaction         46606 non-null  object
dtypes: datetime64[ns](3), float64(5), object(13)
memory usage: 7.5+ MB
```

[77]:
```
data.describe()
```

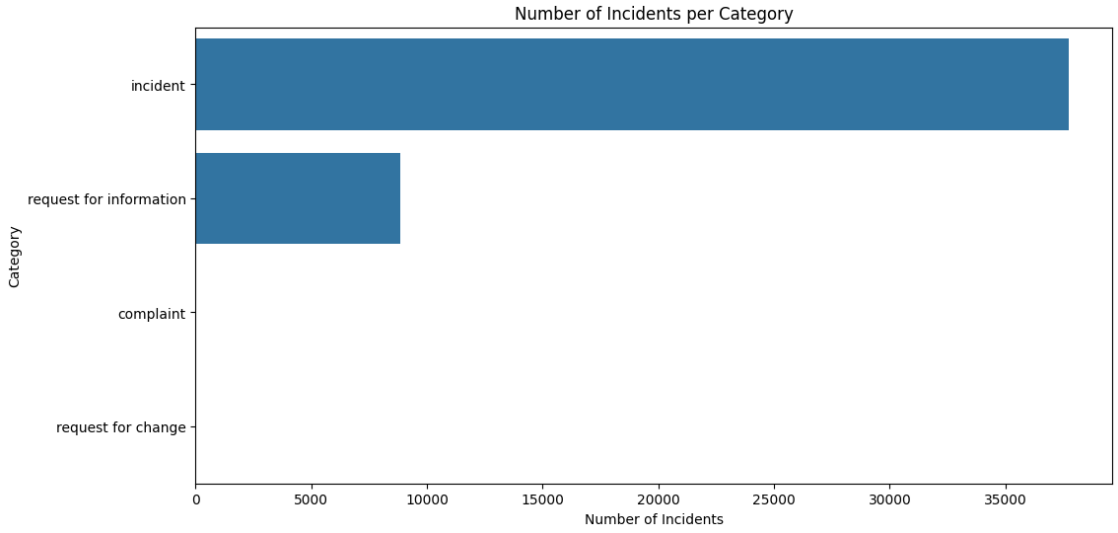
[77]:
```
           Priority    number_cnt  No_of_Reassignments  \
count  46606.000000  46606.000000         46606.000000
mean       4.216989      0.499658             0.954620
min        2.500000      0.000023             0.000000
25%        4.000000      0.248213             0.000000
50%        4.000000      0.500269             0.000000
75%        5.000000      0.749094             2.000000
max        5.000000      0.999997             5.000000
std        0.674287      0.288634             1.458013

                           Open_Time                  Resolved_Time  \
count                          18612                          18333
mean   2013-12-10 03:34:32.927143680  2013-12-15 14:00:25.442644224
min              2012-01-10 10:49:00            2013-01-10 06:45:00
25%              2013-06-11 11:40:45            2013-06-11 13:59:00
50%              2013-11-12 13:00:30            2013-12-11 07:52:00
75%              2014-06-02 18:22:30            2014-06-03 14:26:00
max              2014-12-03 22:58:00            2014-12-03 17:56:00
std                              NaN                            NaN

                          Close_Time  Handle_Time_hrs  \
count                          18333       46606.000
mean   2013-12-15 14:00:49.795450880           2.625
min              2013-01-10 06:45:00           2.625
25%              2013-06-11 13:59:00           2.625
50%              2013-12-11 07:53:00           2.625
75%              2014-06-03 14:27:00           2.625
max              2014-12-03 17:56:00           2.625
std                              NaN           0.000

       No_of_Related_Interactions
count                     46606.0
mean                          1.0
min                           1.0
25%                           1.0
50%                           1.0
75%                           1.0
max                           1.0
std                           0.0
```

[78]:
```
data.isna().sum()
```

[78]:
```
CI_Name                     0
CI_Cat                      0
CI_Subcat                   0
WBS                         0
Incident_ID                 0
```

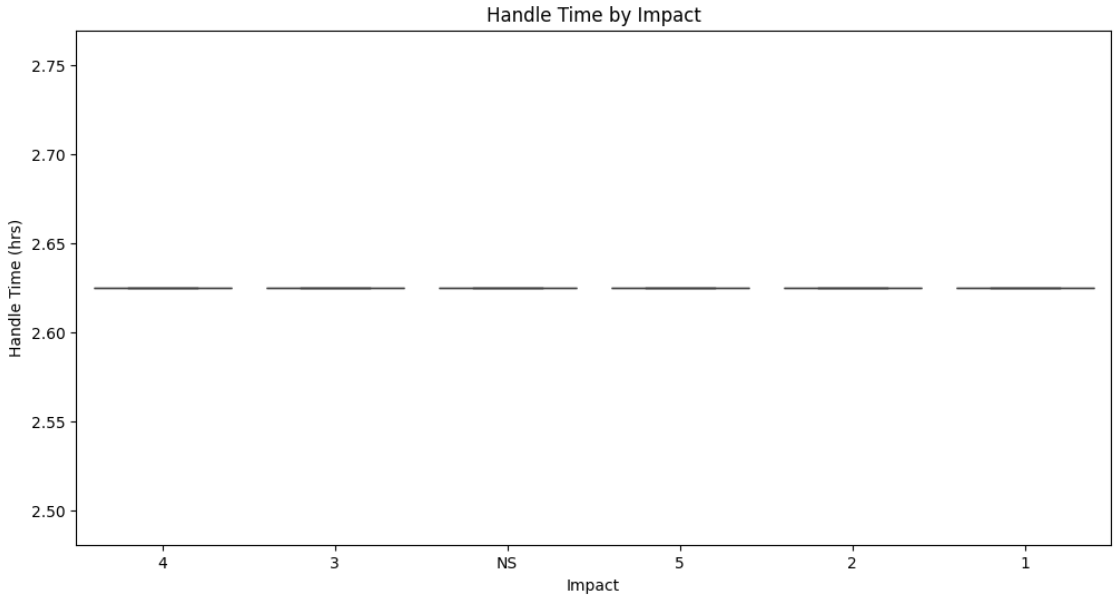
```
Status                          0
Impact                          0
Urgency                         0
Priority                        0
number_cnt                      0
Category                        0
KB_number                       0
Alert_Status                    0
No_of_Reassignments             0
Open_Time                   27994
Resolved_Time               28273
Close_Time                  28273
Handle_Time_hrs                 0
Closure_Code                    0
No_of_Related_Interactions      0
Related_Interaction             0
dtype: int64
```

[79]:
```
# Bar Plot for Category Counts
plt.figure(figsize=(12, 6))
sns.countplot(y='Category', data=data, order=data['Category'].value_counts().
 ↪index)
plt.title('Number of Incidents per Category')
plt.xlabel('Number of Incidents')
plt.ylabel('Category')
plt.show()
```



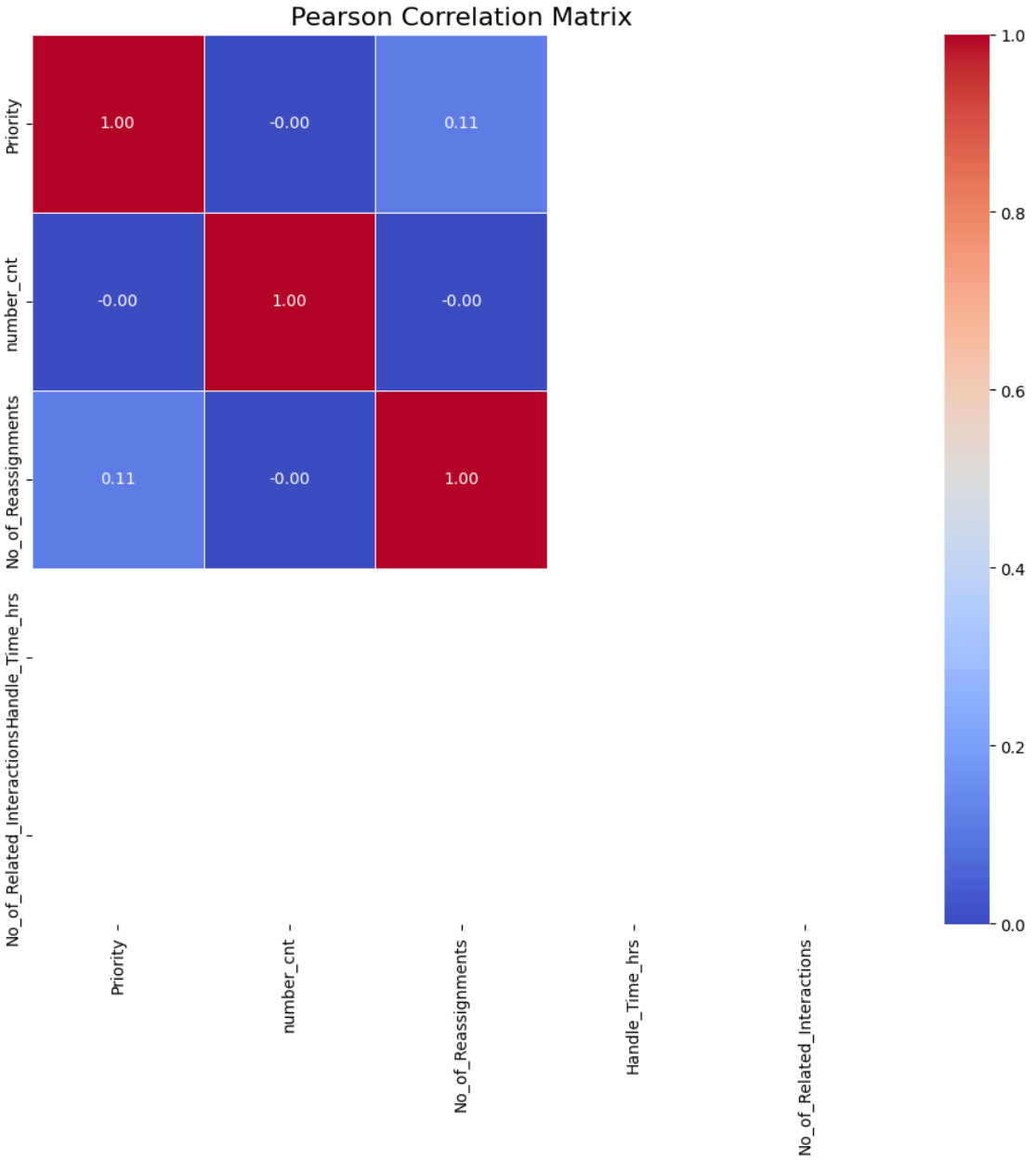
[80]:
```
# Box Plot for Handle Time by Impact
plt.figure(figsize=(12, 6))
```

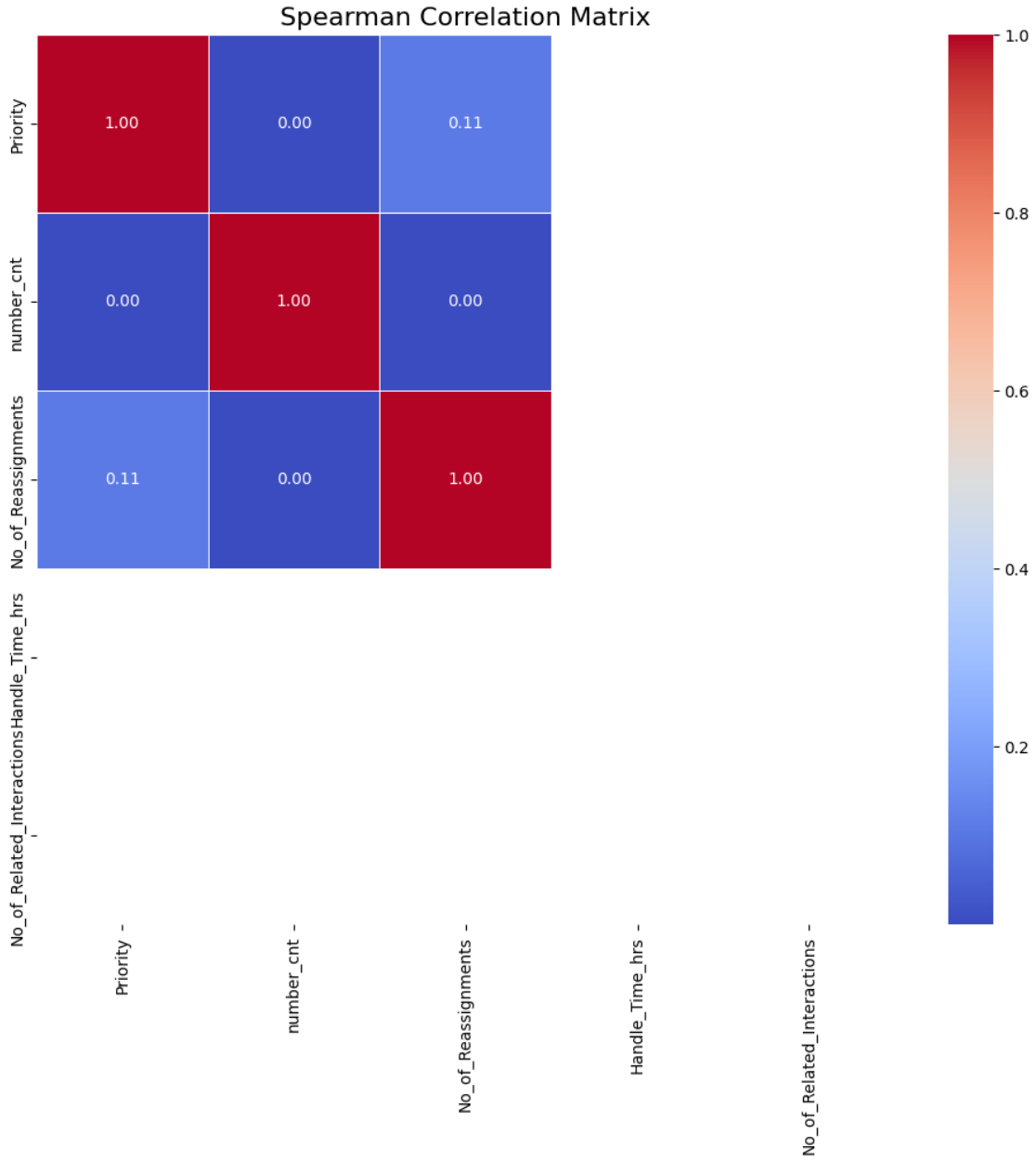
```
sns.boxplot(x='Impact', y='Handle_Time_hrs', data=data)
plt.title('Handle Time by Impact')
plt.xlabel('Impact')
plt.ylabel('Handle Time (hrs)')
plt.show()
```

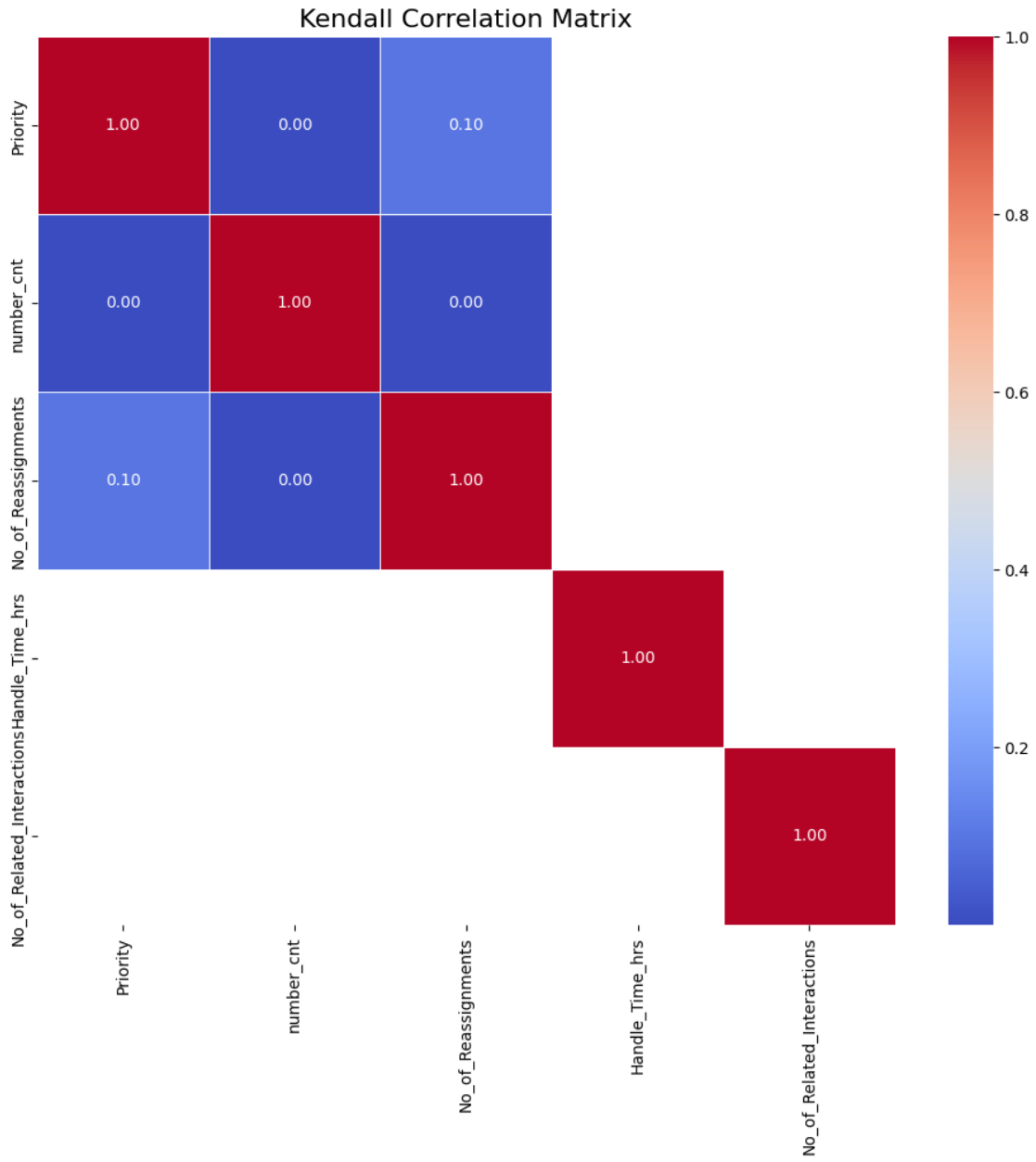


[81]:
```
# Existing Code: Select numerical columns and plot Pearson correlation
numerical_cols = data.select_dtypes(include=['float64', 'int64']).columns
corr_matrix = data[numerical_cols].corr(method='pearson')

plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Pearson Correlation Matrix', fontsize=16)
plt.show()

# Additional Code: Compute Spearman and Kendall correlations
spearman_corr = data[numerical_cols].corr(method='spearman')
kendall_corr = data[numerical_cols].corr(method='kendall')

# Plot Spearman Correlation Matrix
plt.figure(figsize=(12, 10))
sns.heatmap(spearman_corr, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.
↪5)
plt.title('Spearman Correlation Matrix', fontsize=16)
plt.show()

# Plot Kendall Correlation Matrix
```

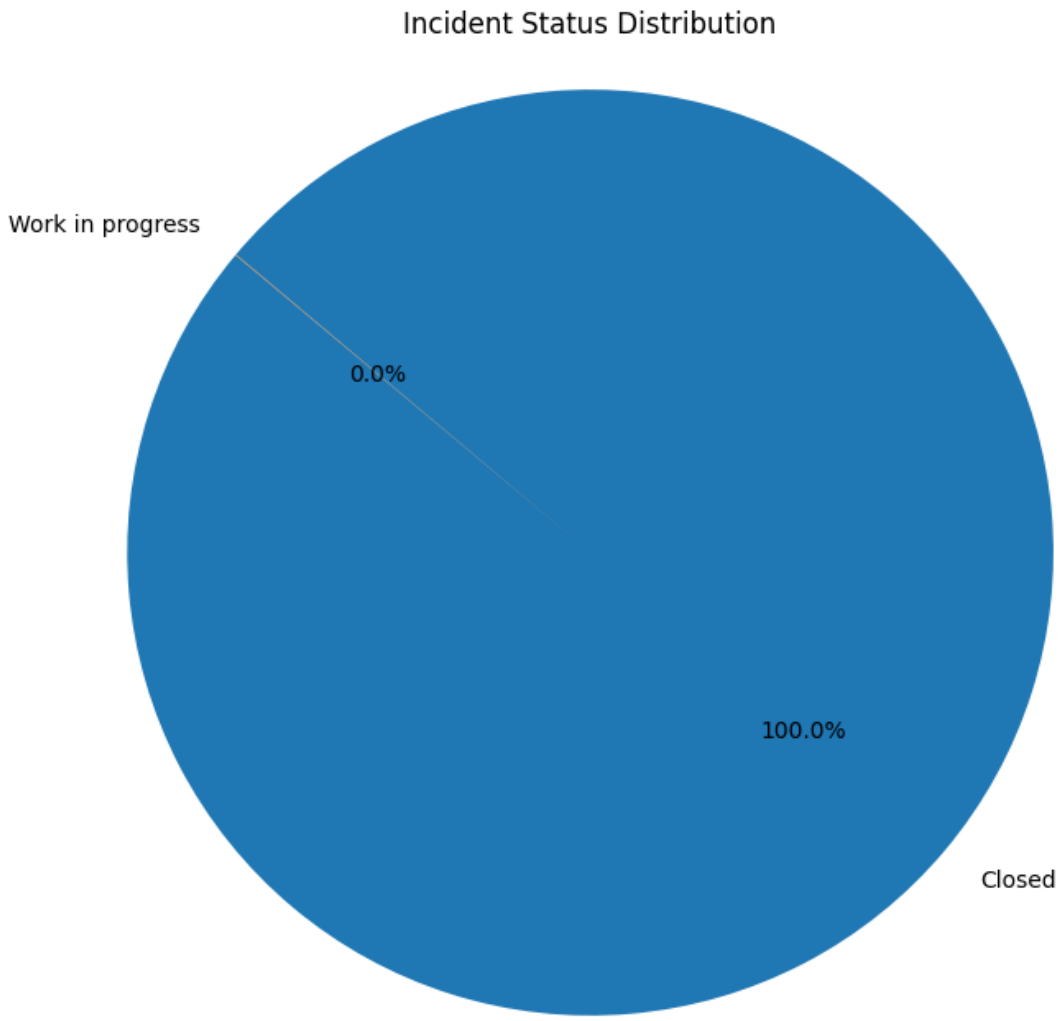
```
plt.figure(figsize=(12, 10))
sns.heatmap(kendall_corr, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Kendall Correlation Matrix', fontsize=16)
plt.show()
```




Spearman Correlation Matrix
Priority
number_cnt
No_of_Reassignments
Handle_Time_hrs
No_of_Related_Interactions
1.00
0.00
0.11
0.00
1.00
0.00
0.11
0.00
1.00
Priority
number_cnt
No_of_Reassignments
Handle_Time_hrs
No_of_Related_Interactions
1.0
0.8
0.6
0.4
0.2

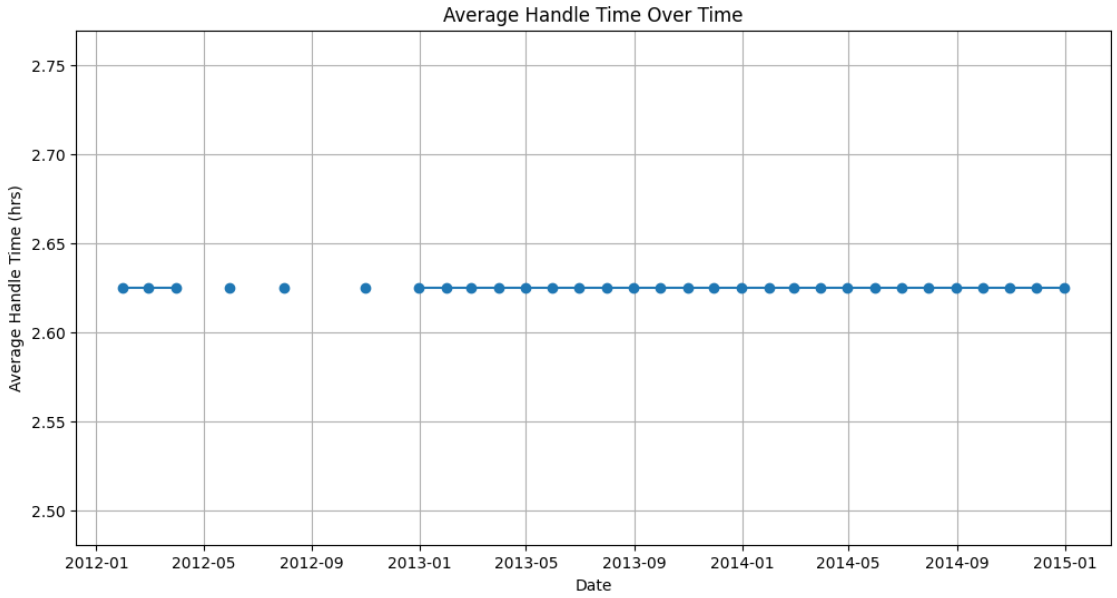


[82]:
```
# Pie Chart for Incident Status Distribution
plt.figure(figsize=(8, 8))
status_counts = data['Status'].value_counts()
plt.pie(status_counts, labels=status_counts.index, autopct='%1.1f%%',␣
↪startangle=140)
plt.title('Incident Status Distribution')
plt.axis('equal')
plt.show()
```



[83]:
```
# Convert 'Open_Time' to datetime
data['Open_Time'] = pd.to_datetime(data['Open_Time'])

# Group by month and calculate the mean Handle_Time_hrs
monthly_avg_handle_time = data.groupby(pd.Grouper(key='Open_Time',␣
 ↪freq='M'))['Handle_Time_hrs'].mean()

# Plotting
plt.figure(figsize=(12, 6))
plt.plot(monthly_avg_handle_time.index, monthly_avg_handle_time.values,␣
 ↪marker='o')
plt.title('Average Handle Time Over Time')
plt.xlabel('Date')
plt.ylabel('Average Handle Time (hrs)')
```

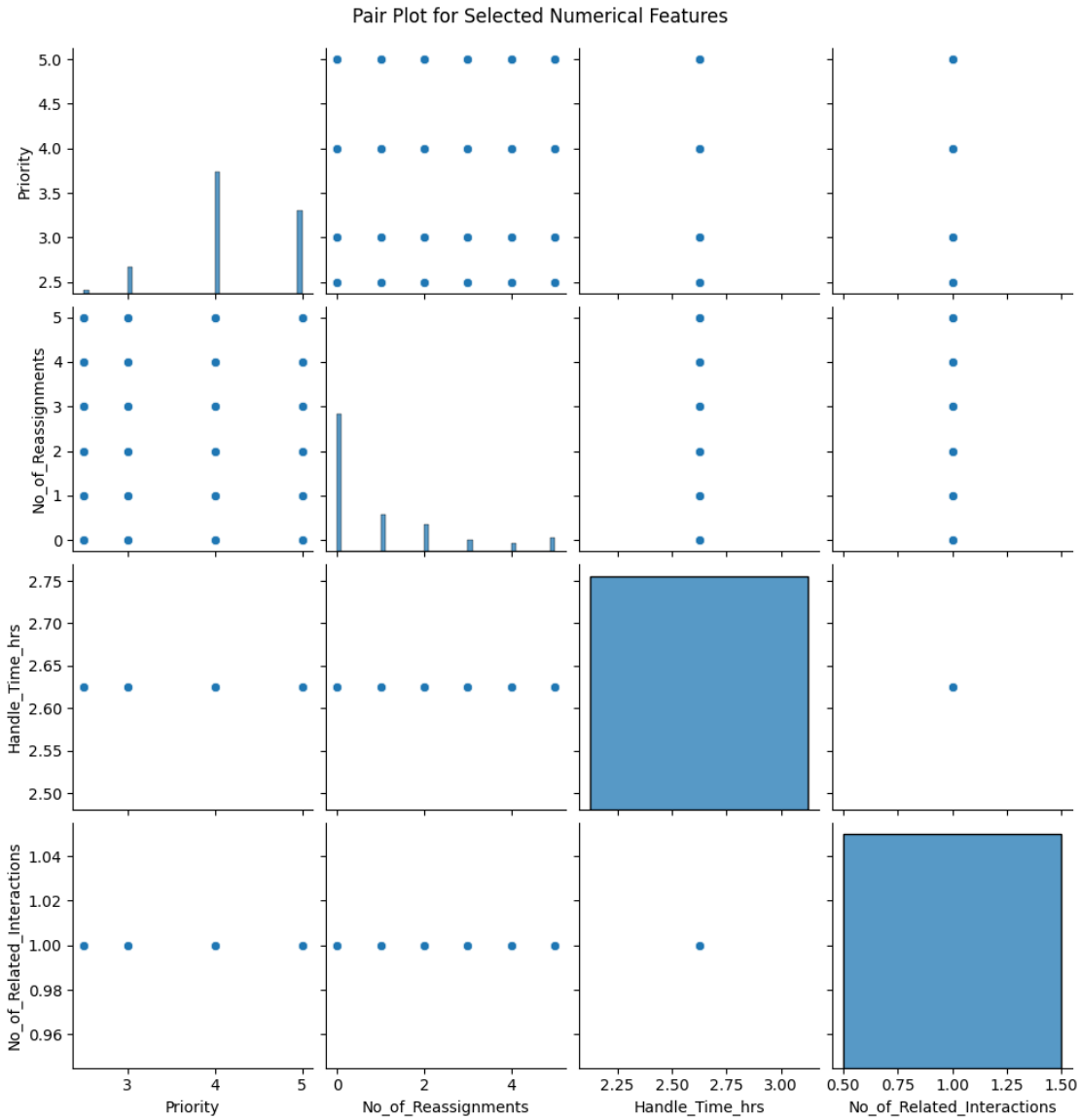
```
plt.grid(True)
plt.show()
```



[84]:
```
data['Open_Time']
```

[84]:
```
0       2012-05-02 13:32:00
1       2012-12-03 15:44:00
2                       NaT
3                       NaT
4       2012-10-08 11:01:00
                ...
46601                   NaT
46602                   NaT
46603                   NaT
46604                   NaT
46605                   NaT
Name: Open_Time, Length: 46606, dtype: datetime64[ns]
```

[85]:
```
# Pair Plot for Selected Numerical Features
selected_columns = ['Priority', 'No_of_Reassignments', 'Handle_Time_hrs',␣
 ↪'No_of_Related_Interactions']
sns.pairplot(data[selected_columns])
plt.suptitle('Pair Plot for Selected Numerical Features', y=1.02)
plt.show()
```



[86]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46606 entries, 0 to 46605
Data columns (total 21 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   CI_Name                     46606 non-null  object
 1   CI_Cat                      46606 non-null  object
 2   CI_Subcat                   46606 non-null  object
 3   WBS                         46606 non-null  object
 4   Incident_ID                 46606 non-null  object
```

```
 5   Status                       46606 non-null  object
 6   Impact                       46606 non-null  object
 7   Urgency                      46606 non-null  object
 8   Priority                     46606 non-null  float64
 9   number_cnt                   46606 non-null  float64
 10  Category                     46606 non-null  object
 11  KB_number                    46606 non-null  object
 12  Alert_Status                 46606 non-null  object
 13  No_of_Reassignments          46606 non-null  float64
 14  Open_Time                    18612 non-null  datetime64[ns]
 15  Resolved_Time                18333 non-null  datetime64[ns]
 16  Close_Time                   18333 non-null  datetime64[ns]
 17  Handle_Time_hrs              46606 non-null  float64
 18  Closure_Code                 46606 non-null  object
 19  No_of_Related_Interactions   46606 non-null  float64
 20  Related_Interaction          46606 non-null  object
dtypes: datetime64[ns](3), float64(5), object(13)
memory usage: 7.5+ MB
```

[87]:
```
from sklearn.preprocessing import LabelEncoder

# Initialize the LabelEncoder
label_encoder = LabelEncoder()
for column in data.columns:
    if data[column].dtype == 'object':  # Check if the column is categorical
        data[column] = data[column].astype(str)
        data[column] = label_encoder.fit_transform(data[column])
        print(f'Column "{column}" has been encoded.')

print("\nEncoded DataFrame:")
print(data)
```

```
Column "CI_Name" has been encoded.
Column "CI_Cat" has been encoded.
Column "CI_Subcat" has been encoded.
Column "WBS" has been encoded.
Column "Incident_ID" has been encoded.
Column "Status" has been encoded.
Column "Impact" has been encoded.
Column "Urgency" has been encoded.
Column "Category" has been encoded.
Column "KB_number" has been encoded.
Column "Alert_Status" has been encoded.
Column "Closure_Code" has been encoded.
Column "Related_Interaction" has been encoded.

Encoded DataFrame:
       CI_Name  CI_Cat  CI_Subcat  WBS  Incident_ID  Status  Impact  Urgency  \
```

```
0         2741      12          58  137            0       0       3        3
1         2863       2          58   70            1       0       2        2
2          990       2          10   74            2       0       5        2
3         2863       2          58   70            3       0       3        3
4         2863       2          58   70            4       0       3        3
…          …     …         …   …          …     …      …       …
46601     2455       2          45   59        46601       0       3        3
46602     2453       2          45   59        46602       0       3        3
46603     1096       4          21   73        46603       0       4        4
46604     2834       2          58   59        46604       0       3        3
46605      674       7           6  218        46605       0       2        2

       Priority  number_cnt  Category  KB_number  Alert_Status  \
0           4.0    0.601292         1        413             0
1           3.0    0.415050         1        456             0
2           4.0    0.517551         3        244             0
3           4.0    0.642927         1        456             0
4           4.0    0.345258         1        456             0
…          …          …       …         …           …
46601       4.0    0.231896         1       1025             0
46602       4.0    0.805153         1       1821             0
46603       5.0    0.917466         1        225             0
46604       4.0    0.701278         1       1003             0
46605       3.0    0.902320         1        134             0

       No_of_Reassignments           Open_Time       Resolved_Time  \
0                      5.0 2012-05-02 13:32:00 2013-04-11 13:50:00
1                      5.0 2012-12-03 15:44:00 2013-02-12 12:36:00
2                      3.0                 NaT                 NaT
3                      5.0                 NaT                 NaT
4                      2.0 2012-10-08 11:01:00 2013-08-11 13:55:00
…                     …                 …                 …
46601                  0.0                 NaT                 NaT
46602                  0.0                 NaT                 NaT
46603                  0.0                 NaT                 NaT
46604                  0.0                 NaT                 NaT
46605                  0.0                 NaT                 NaT

               Close_Time  Handle_Time_hrs  Closure_Code  \
0     2013-04-11 13:51:00            2.625             6
1     2013-02-12 12:36:00            2.625            10
2                     NaT            2.625             4
3                     NaT            2.625             5
4     2013-08-11 13:55:00            2.625             6
…                    …              …           …
46601                 NaT            2.625             6
46602                 NaT            2.625            12
46603                 NaT            2.625             1
```

```
46604                  NaT            2.625            10
46605                  NaT            2.625             1

       No_of_Related_Interactions  Related_Interaction
0                             1.0                    2
1                             1.0                    3
2                             1.0                    4
3                             1.0                    5
4                             1.0                    6
…                             …                      …
46601                         1.0                43057
46602                         1.0                43052
46603                         1.0                43053
46604                         1.0                43054
46605                         1.0                43059

[46606 rows x 21 columns]
```

[88]:
```
data.to_csv('preprocessed data.csv',index=False)
```

[89]:
```
data1=pd.read_csv('preprocessed data.csv')
```

[90]:
```
data1.head()
```

[90]:

	CI_Name	CI_Cat	CI_Subcat	WBS	Incident_ID	Status	Impact	Urgency
0	2741	12	58	137	0	0	3	3
1	2863	2	58	70	1	0	2	2
2	990	2	10	74	2	0	5	2
3	2863	2	58	70	3	0	3	3
4	2863	2	58	70	4	0	3	3

	Priority	number_cnt	Category	KB_number	Alert_Status
0	4.0	0.601292	1	413	0
1	3.0	0.415050	1	456	0
2	4.0	0.517551	3	244	0
3	4.0	0.642927	1	456	0
4	4.0	0.345258	1	456	0

	No_of_Reassignments	Open_Time	Resolved_Time
0	5.0	2012-05-02 13:32:00	2013-04-11 13:50:00
1	5.0	2012-12-03 15:44:00	2013-02-12 12:36:00
2	3.0	NaN	NaN
3	5.0	NaN	NaN
4	2.0	2012-10-08 11:01:00	2013-08-11 13:55:00

	Close_Time	Handle_Time_hrs	Closure_Code
0	2013-04-11 13:51:00	2.625	6

```
1  2013-02-12 12:36:00            2.625            10
2                  NaN            2.625             4
3                  NaN            2.625             5
4  2013-08-11 13:55:00            2.625             6

   No_of_Related_Interactions  Related_Interaction
0                         1.0                    2
1                         1.0                    3
2                         1.0                    4
3                         1.0                    5
4                         1.0                    6
```

[91]:
```
data1.isnull().sum()
```

[91]:
```
CI_Name                         0
CI_Cat                          0
CI_Subcat                       0
WBS                             0
Incident_ID                     0
Status                          0
Impact                          0
Urgency                         0
Priority                        0
number_cnt                      0
Category                        0
KB_number                       0
Alert_Status                    0
No_of_Reassignments             0
Open_Time                   27994
Resolved_Time               28273
Close_Time                  28273
Handle_Time_hrs                 0
Closure_Code                    0
No_of_Related_Interactions      0
Related_Interaction             0
dtype: int64
```

[92]:
```
data1.shape
```

[92]:
```
(46606, 21)
```

[93]:
```
unique_values = sorted(data1['Priority'].dropna().unique())
mapping = {value: i for i, value in enumerate(unique_values, start=1)}

# Add mapping for NaN values if needed
mapping[np.nan] = -1  # Replace -1 with any other placeholder value if required
```

```
# Replace values in the series
data1['Priority'] = data1['Priority'].map(mapping).fillna(-1).astype(int)  # Or␣
 ↪replace -1 with another placeholder




features = data[[
    "Impact",
    "Urgency",
    "Status",
    "No_of_Reassignments",
    "Handle_Time_hrs",
    "CI_Name",
    "CI_Cat",
    "CI_Subcat",
    "Alert_Status",
    "Category",
    "Closure_Code",
    "Related_Interaction"
]]  # Drop the target variable
target = data['Priority']  # Target variable
```

[94]:
```
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(features, target,␣
 ↪test_size=0.3, random_state=42)

print(X_train)
print(X_test)
print(y_train)
print(y_test)
```

```
       Impact  Urgency  Status  No_of_Reassignments  Handle_Time_hrs  CI_Name  \
24242       4        4       0                  1.0            2.625     2733
9410        3        3       0                  0.0            2.625     2863
24211       3        3       0                  0.0            2.625     2056
43240       3        3       0                  0.0            2.625     2725
36954       2        2       0                  1.0            2.625     2278
…         …      …       …                    …              …       …
11284       3        3       0                  5.0            2.625     2879
44732       3        3       0                  0.0            2.625      920
38158       3        3       0                  1.0            2.625     2487
860         3        3       0                  1.0            2.625     2584
15795       2        2       0                  0.0            2.625     2278
```

```
       CI_Cat  CI_Subcat  Alert_Status  Category  Closure_Code  \
24242      12         58             0         1             6
9410        2         58             0         1            10
24211       6         27             0         1             1
43240      12         58             0         1            10
36954       2         45             0         1            12
…        …         …           …        …             …
11284       2         58             0         1            10
44732       4          9             0         1            11
38158       2         45             0         1             6
860         2         45             0         1            10
15795       2         45             0         1            12

       Related_Interaction
24242                    0
9410                  8524
24211                22078
43240                39845
36954                33965
…                    …
11284                10344
44732                    0
38158                35126
860                    665
15795                14421

[32624 rows x 12 columns]
       Impact  Urgency  Status  No_of_Reassignments  Handle_Time_hrs  CI_Name  \
10214       4        4       0                  2.0            2.625      996
17442       4        4       0                  5.0            2.625     2816
41471       3        3       0                  1.0            2.625     2684
46248       4        4       0                  0.0            2.625     2684
9672        1        1       0                  1.0            2.625     2815
…        …       …     …                  …              …       …
34350       3        3       0                  0.0            2.625      991
20015       3        3       0                  0.0            2.625     2725
31287       3        3       0                  4.0            2.625     2869
2435        1        1       0                  0.0            2.625     2398
35216       4        3       0                  0.0            2.625      998

       CI_Cat  CI_Subcat  Alert_Status  Category  Closure_Code  \
10214       2         10             0         1             1
17442       2         58             0         1            10
41471      12         45             0         1             6
46248      12         45             0         1            10
9672        2         58             0         1            10
…        …         …           …        …             …
34350       2         10             0         1             6
```

```
20015      12          58              0          1             12
31287       2          58              0          1             13
2435        2          45              0          1             12
35216       2          10              0          1              6

       Related_Interaction
10214                 9223
17442                15906
41471                    0
46248                42706
9672                     0
…                        …
34350                31547
20015                18243
31287                    0
2435                  2059
35216                32392

[13982 rows x 12 columns]
24242    5.0
9410     4.0
24211    4.0
43240    4.0
36954    3.0
        …
11284    4.0
44732    4.0
38158    4.0
860      4.0
15795    3.0
Name: Priority, Length: 32624, dtype: float64
10214    5.0
17442    5.0
41471    4.0
46248    5.0
9672     2.5
        …
34350    4.0
20015    4.0
31287    4.0
2435     2.5
35216    4.0
Name: Priority, Length: 13982, dtype: float64
```

[95]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

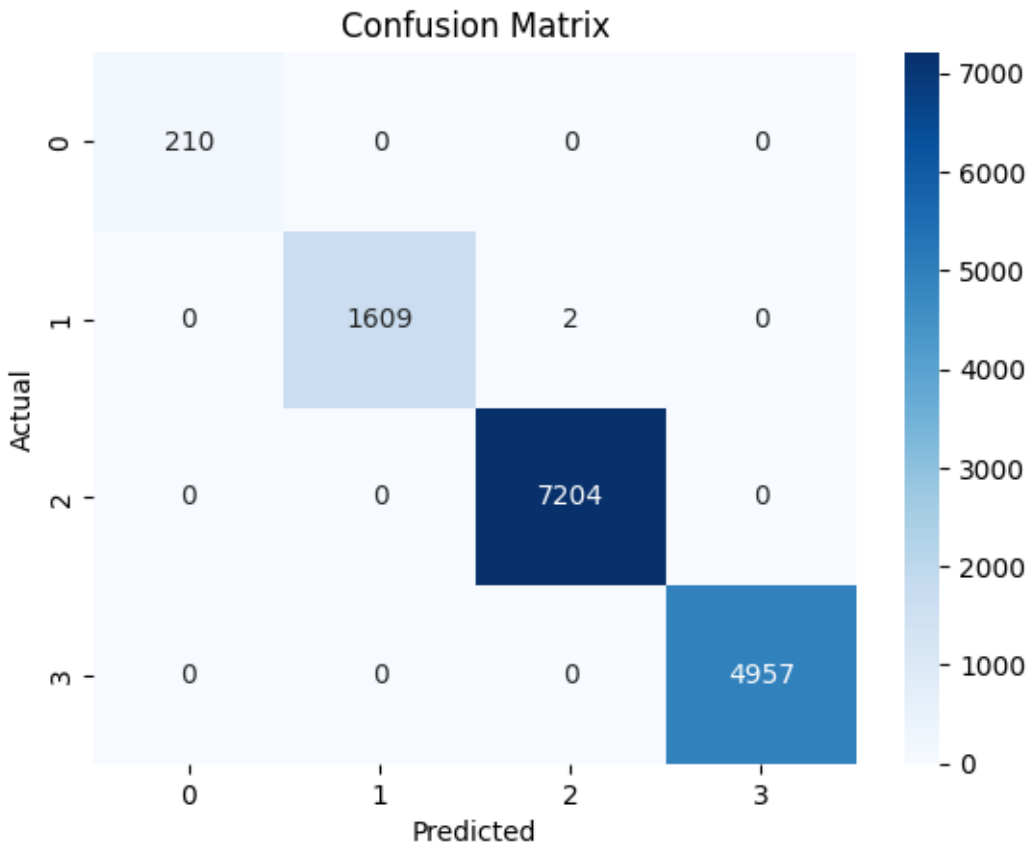
```
# Fit the encoder and transform the labels
y_train_encoded = le.fit_transform(y_train)
y_test_encoded = le.transform(y_test)
print(y_train_encoded)
print(y_test_encoded)
```

```
[3 2 2 … 2 2 1]
[3 3 2 … 2 0 2]
```

[96]:
```
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import accuracy_score, precision_score, recall_score



# Initialize the model
rf = RandomForestClassifier(random_state=42)

# Define parameter grid for tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Use GridSearchCV to find the best parameters
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5,␣
 ↪scoring='accuracy', n_jobs=-1, verbose=2)
grid_search.fit(X_train, y_train_encoded)

# Best parameters and model
best_rf = grid_search.best_estimator_
print("Best Parameters:", grid_search.best_params_)



# Predictions and evaluation
y_pred = best_rf.predict(X_test)
print("y_test dtype:", y_test.dtype)
print("y_pred dtype:", y_pred.dtype)
print("y_test unique values:", np.unique(y_test))
print("y_pred unique values:", np.unique(y_pred))

# Print classification report and confusion matrix
print("Confusion Matrix:\n", confusion_matrix(y_test_encoded, y_pred))
print("Classification Report:\n", classification_report(y_test_encoded, y_pred))
```

```
print("Accuracy:", accuracy_score(y_test_encoded, y_pred))
precision = precision_score(y_test_encoded, y_pred, average='weighted')
recall = recall_score(y_test_encoded, y_pred, average='weighted')
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
# Plot confusion matrix
cm = confusion_matrix(y_test_encoded, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

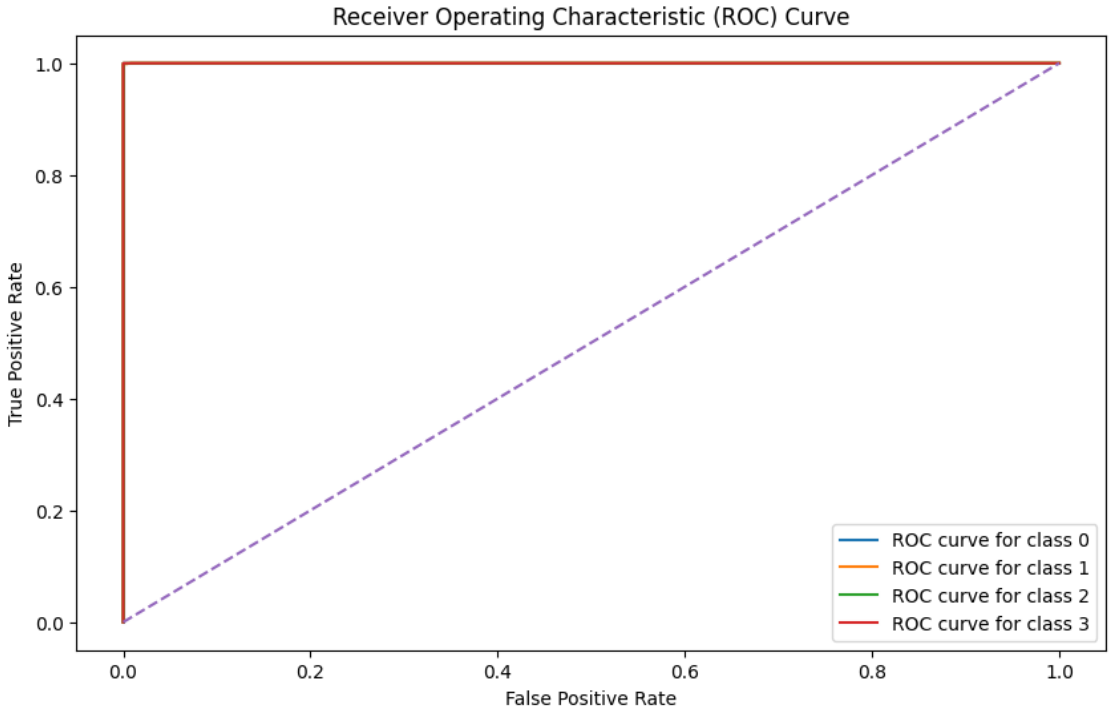
```
Fitting 5 folds for each of 81 candidates, totalling 405 fits
Best Parameters: {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split':
10, 'n_estimators': 200}
y_test dtype: float64
y_pred dtype: int64
y_test unique values: [2.5 3.  4.  5. ]
y_pred unique values: [0 1 2 3]
Confusion Matrix:
 [[ 210    0    0    0]
 [   0 1609    2    0]
 [   0    0 7204    0]
 [   0    0    0 4957]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       210
           1       1.00      1.00      1.00      1611
           2       1.00      1.00      1.00      7204
           3       1.00      1.00      1.00      4957

    accuracy                           1.00     13982
   macro avg       1.00      1.00      1.00     13982
weighted avg       1.00      1.00      1.00     13982

Accuracy: 0.9998569589472178
Precision: 1.00
Recall: 1.00
```



[97]:
```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import roc_auc_score, roc_curve
import numpy as np
import matplotlib.pyplot as plt

# Cross-validation score
cv_scores = cross_val_score(best_rf, X_train, y_train_encoded, cv=5,␣
 ↪scoring='accuracy')
print(f"Cross-Validation Accuracy Scores: {cv_scores}")
print(f"Mean CV Accuracy: {np.mean(cv_scores)}")

# ROC-AUC score and curve
# Specify the 'multi_class' parameter as 'ovr' (one-vs-rest)
y_prob = best_rf.predict_proba(X_test)
roc_auc = roc_auc_score(y_test_encoded, y_prob, multi_class='ovr')  # Specify␣
 ↪the multi_class parameter

# Compute ROC curve for each class
fpr = {}
tpr = {}
```

```
for i in range(y_prob.shape[1]):
    fpr[i], tpr[i], _ = roc_curve(y_test_encoded, y_prob[:, i], pos_label=i)

# Plot ROC Curve for each class
plt.figure(figsize=(10, 6))
for i in range(y_prob.shape[1]):
    plt.plot(fpr[i], tpr[i], label=f'ROC curve for class {i}')
plt.plot([0, 1], [0, 1], linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()
```

```
Cross-Validation Accuracy Scores: [1.         1.         1.         1.
0.99984672]
Mean CV Accuracy: 0.9999693439607602
```



[98]:
```
import pickle

# Save the model to a file
model_filename = 'priority_prediction_model.pkl'
with open(model_filename, 'wb') as file:
    pickle.dump(grid_search, file)
```

```
# Load the model (for deployment or later use)
with open(model_filename, 'rb') as file:
    loaded_model = pickle.load(file)

Impact = int(input("enter the value between 1 to 6 :- "))

Urgency =int(input("enter the value between 1 to 11 :- "))

Status=int(input("enter the value between 1 or 2 :- "))

No_of_Reassignments=int(input("enter the value between 1 to 42 :- "))

Handle_Time_hrs = int(input("enter the value it must be  10 digit  :- "))

CI_Name = int(input("enter the value between 1 to 3019 :- "))

CI_Cat = int(input("enter the value between 1 to 13 :- "))

CI_Subcat = int(input("enter the value between 1 to 65 :- "))

Alert_Status = int(input("enter the value 1 :- "))

Category = int(input("enter the value between 1 to 4 :- "))

Closure_Code = int(input("enter the value between 1 to 15 :- "))

Related_Interaction = int(input("enter the value between 1 to 43060 :- "))


# Example prediction
example_data = [[Impact,Urgency,
Status,
No_of_Reassignments,
Handle_Time_hrs,
CI_Name,
CI_Cat,
CI_Subcat,
Alert_Status,
Category,
Closure_Code,
Related_Interaction]]  # Example feature values
predicted_priority = loaded_model.predict(example_data)
print(f"Predicted Priority: {predicted_priority[0]}")
for i in example_data:
    print(i)
```

```
Predicted Priority: 1
[2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

[ ]: