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

# For feature selection
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel

# For statistical analysis
from scipy import stats


df=pd.read_csv('file1.csv')
df
```

	time	device_address	physical_payload	gateway	crc_status	fr
0	2019-03-01T00:00:01.528887066Z	01ae0905	QAUJrgGA3CIF/9yXgSG9G2+SwnOEzEmlHJeTCuE=	00000f0c210721f2	1	86
1	2019-03-01T00:00:01.528930845Z	01ae0905	QAUJrgGA3CIF/9yXgSG9G2+SwnOEzEmlHJeTCuE=	00000f0c210721f2	1	86
2	2019-03-01T00:00:01.822131604Z	00c2d5cc	QMzVwgCA7xoFASlHknHqttrHhvTB4DvnUz/HSvs=	00000f0c210721f2	1	86
3	2019-03-01T00:00:01.822184581Z	00c2d5cc	QMzVwgCA7xoFASlHknHqttrHhvTB4DvnUz/HSvs=	00000f0c210721f2	1	86
4	2019-03-01T00:00:03.924795694Z	-1	NaN	00000f0c210721f2	-1	86
...	...	...	...	...	...	
108420	2019-03-01T23:59:43.726314706Z	10003432	QDI0ABCAug8BmDSRJF3Vv8eR3OH07twjPyKrX0tcqtwwB2...	00000f0c22433141	1	86
108421	2019-03-01T23:59:56.371057987Z	-1	NaN	00000f0c22433141	-1	86
108422	2019-03-01T23:58:19.264275Z	fa2dd4c4	m8TULfqnX/aTLvbsiRvo/jATE+0RiCFoQ2UC8lxemOLVRn...	00800000a0001793	-1	86
108423	2019-03-01T23:57:25.821769386Z	8e98fc05	ewX8ml6/wjnOKqoazwPuba7eeRp+o9fZ7jNLZkRhKh/Pp8...	00800000a0001914	-1	86
108424	2019-03-01T23:58:01.389385243Z	e04f2714	fBQnT+C46c+bkOiIfuCJUzH6MKwg+xHjPh2USgv+67U3nd...	00800000a0001914	-1	86

108425 rows × 15 columns

Start coding or generate with AI.

```
print(df.describe())
```

```
          crc_status     frequency  spreading_factor      bandwidth  \
count  108425.000000  1.084250e+05     108424.000000  108424.000000
mean        0.359041  8.681482e+08          7.986433     125.547619
std         0.933326  3.567392e+05          0.916765       8.255487
min        -1.000000  8.671000e+08          7.000000     125.000000
25%        -1.000000  8.681000e+08          7.000000     125.000000
50%         1.000000  8.681000e+08          8.000000     125.000000
75%         1.000000  8.685000e+08          8.000000     125.000000
max         1.000000  8.688000e+08         12.000000     250.000000

                rssi           snr           size          mtype  \
count  108425.000000  108425.000000  108425.000000  108425.000000
mean     -108.471312      -6.161835      21.315822       8.812184
std         6.019582       6.129328      19.387354      15.755465
min      -123.000000    -128.000000       0.000000      -1.000000
25%      -113.000000     -11.000000       0.000000      -1.000000
50%      -109.000000      -7.000000      29.000000      10.000000
75%      -105.000000      -2.000000      29.000000      10.000000
max       -68.000000      15.000000     255.000000     111.000000

               fcnt          fport
count  108425.000000  108391.000000
mean     9802.214628       4.006467
std     13504.482517      13.112764
min        -1.000000      -1.000000
```

```
25%        -1.000000      -1.000000
50%      5691.000000       5.000000
75%     10554.000000       5.000000
max     65461.000000     255.000000
```

```
# df['time'] = pd.to_datetime(df['time']).astype('int64') // 10**9

# print(df['time'])
```

```
0         1551398401
1         1551398401
2         1551398401
3         1551398401
4         1551398403
             ...
108420    1551484783
108421    1551484796
108422    1551484699
108423    1551484645
108424    1551484681
Name: time, Length: 108425, dtype: int64
```

```
numeric_cols = df.select_dtypes(include='number').columns
print(numeric_cols)
```

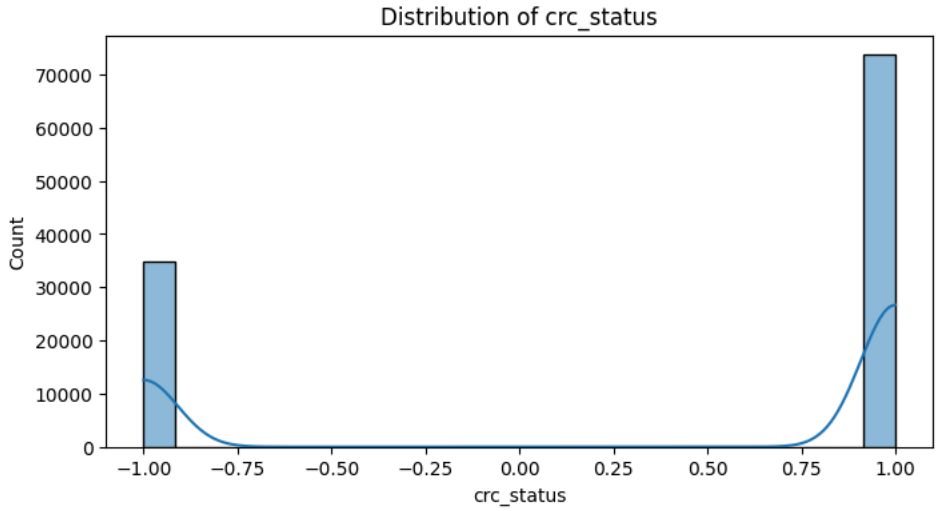
```
Index(['crc_status', 'frequency', 'spreading_factor', 'bandwidth', 'rssi',
       'snr', 'size', 'mtype', 'fcnt', 'fport'],
      dtype='object')
```

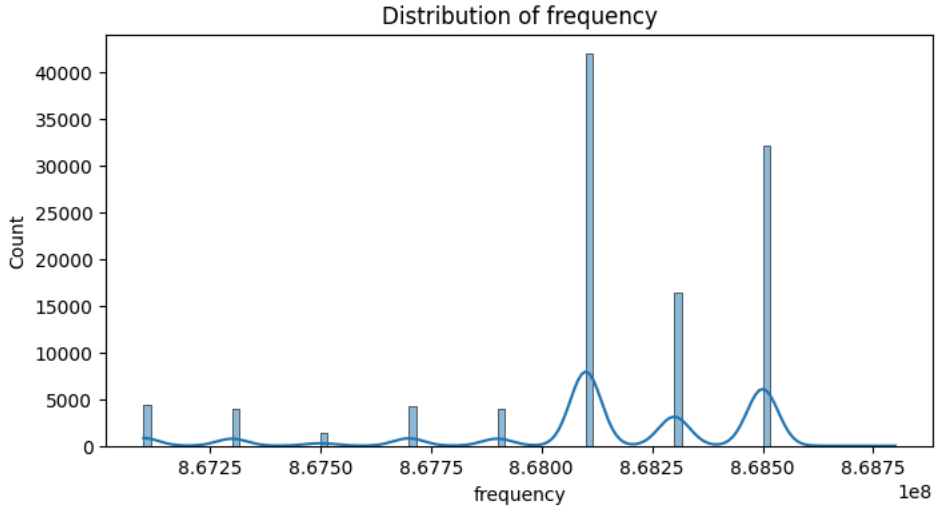
```
selected_cols = numeric_cols.unique()
numeric_df = df[selected_cols]
numeric_df
```

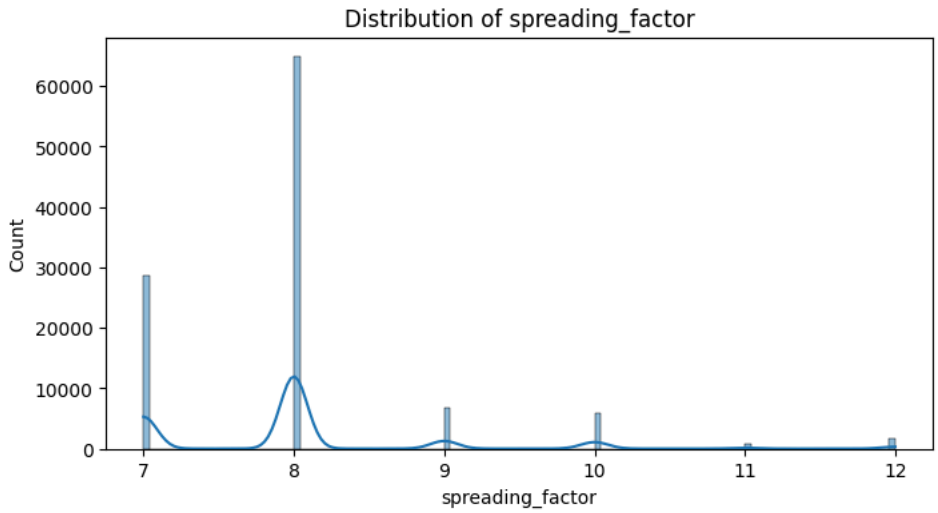
	crc_status	frequency	spreading_factor	bandwidth	rssi	snr	size	mtype	fcnt	fport
0	1	868500000	8.0	125.0	-112	0.0	29	10	8924	5.0
1	1	868500000	8.0	125.0	-113	-9.0	29	10	8924	5.0
2	1	868100000	8.0	125.0	-108	2.0	29	10	6895	5.0
3	1	868100000	8.0	125.0	-101	9.0	29	10	6895	5.0
4	-1	868100000	8.0	125.0	-112	-11.0	0	-1	-1	-1.0
...	...	...	...	...	...	...	...	...	...	...
108420	1	868500000	7.0	125.0	-115	-7.0	42	10	4026	1.0
108421	-1	868100000	8.0	125.0	-116	-13.5	0	-1	-1	-1.0
108422	-1	867700000	7.0	125.0	-114	-10.8	47	100	63071	254.0
108423	-1	867300000	7.0	125.0	-111	-11.0	71	11	14786	217.0
108424	-1	867300000	7.0	125.0	-111	-11.5	168	11	53225	49.0

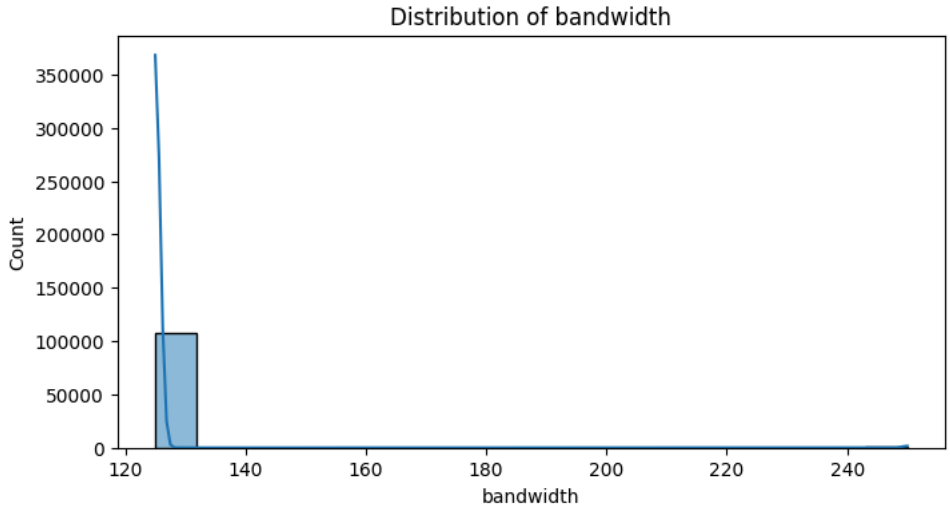
108425 rows × 10 columns

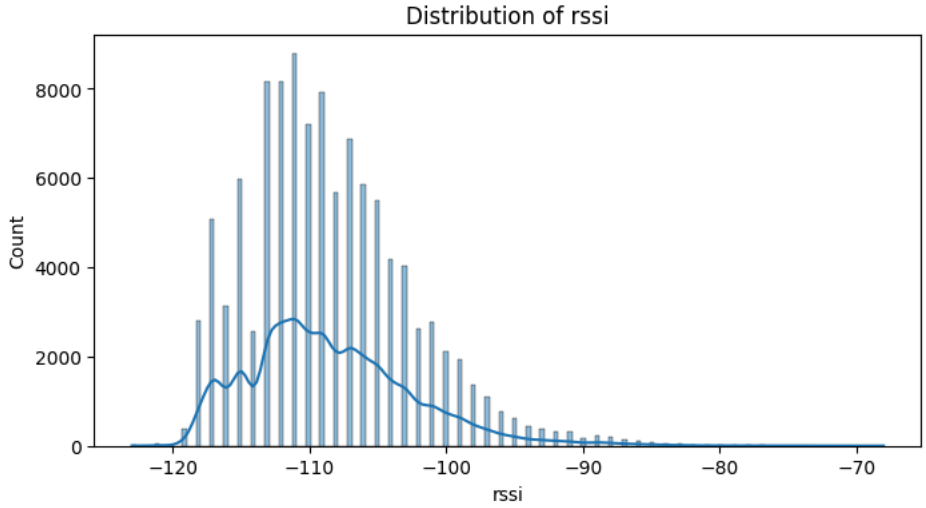
```
# Histograms for numerical features
for col in numeric_df:
    plt.figure(figsize=(8, 4))
    sns.histplot(df[col], kde=True)
    plt.title(f'Distribution of {col}')
    plt.show()
```

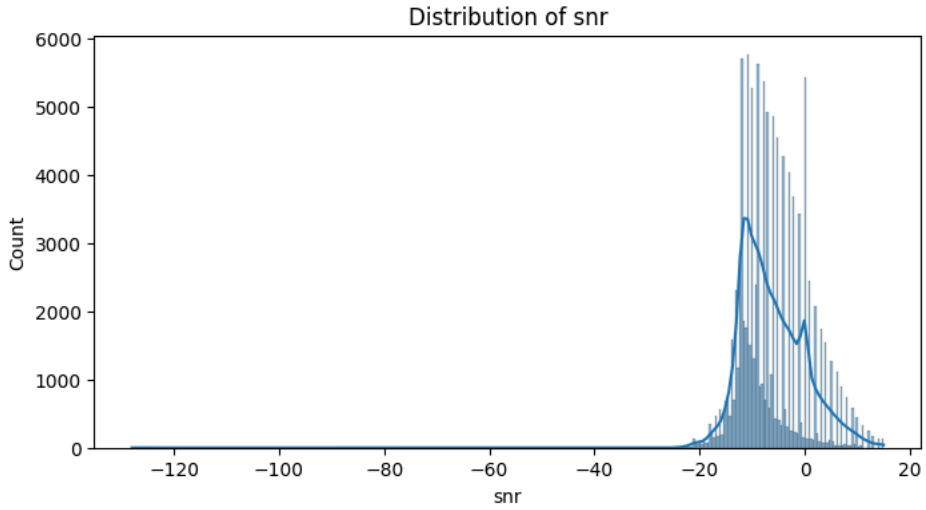

Distribution of crc_status
Count
crc_status
70000
60000
50000
40000
30000
20000
10000
0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

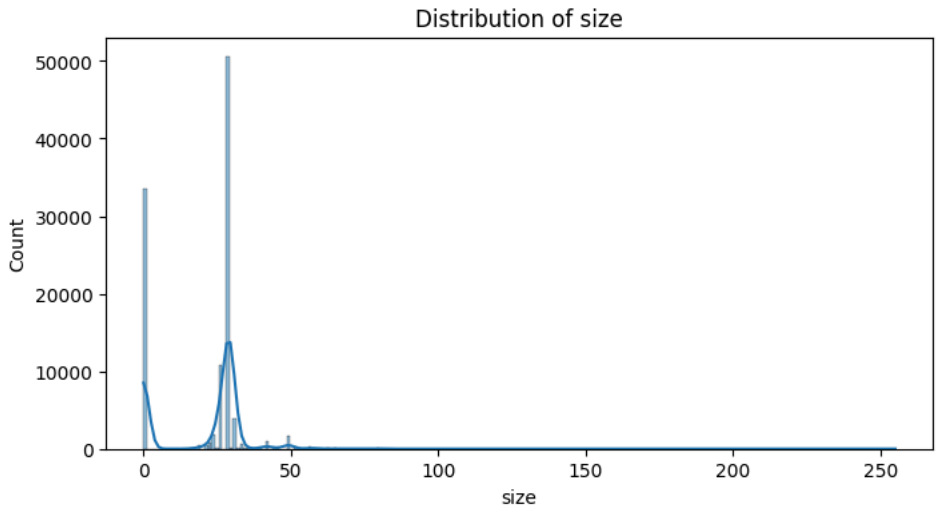

Distribution of frequency
Count
frequency
1e8
40000
35000
30000
25000
20000
15000
10000
5000
0
8.6725
8.6750
8.6775
8.6800
8.6825
8.6850
8.6875

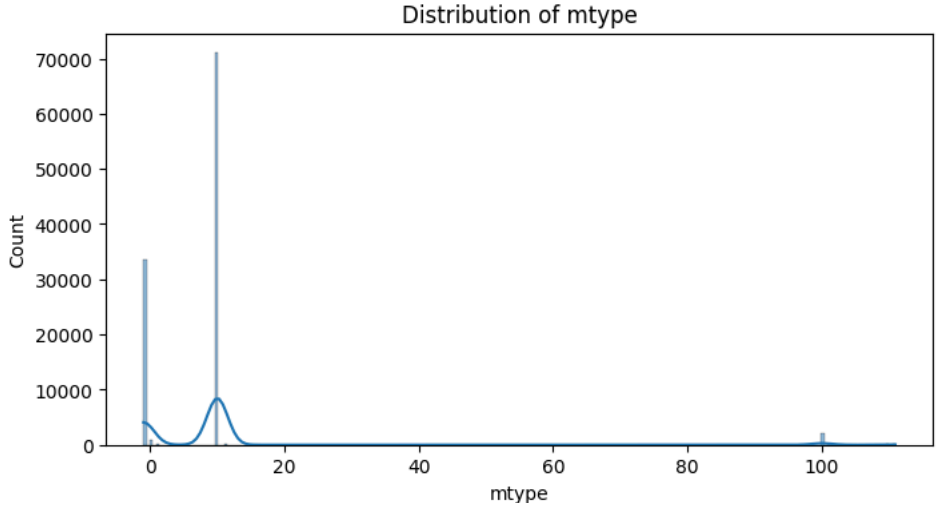

Distribution of spreading_factor
Count
spreading_factor
60000
50000
40000
30000
20000
10000
0
7
8
9
10
11
12

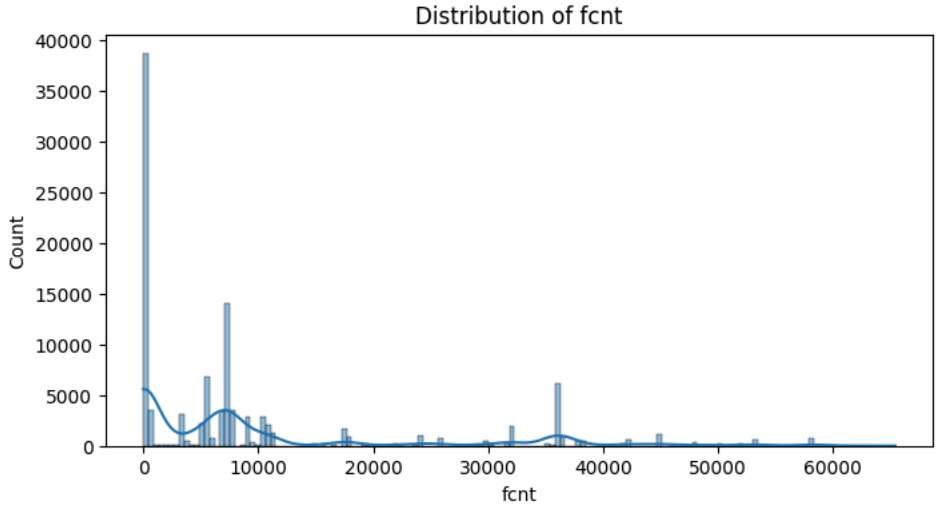

Distribution of bandwidth
Count
bandwidth
350000
300000
250000
200000
150000
100000
50000
0
120
140
160
180
200
220
240

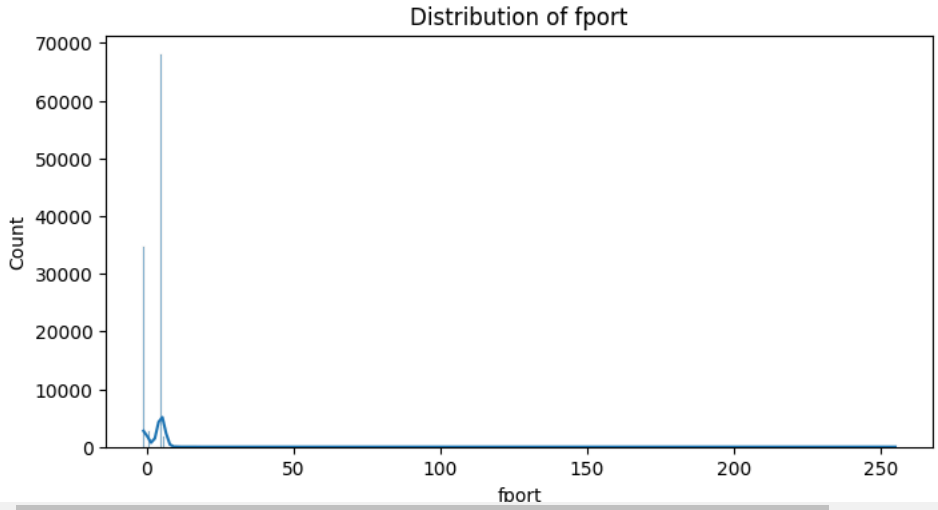

Distribution of rssi
Count
rssi
8000
6000
4000
2000
0
−120
−110
−100
−90
−80
−70


Distribution of snr
Count
snr
6000
5000
4000
3000
2000
1000
0
−120
−100
−80
−60
−40
−20
0
20


Distribution of size
Count
size
50000
40000
30000
20000
10000
0
0
50
100
150
200
250

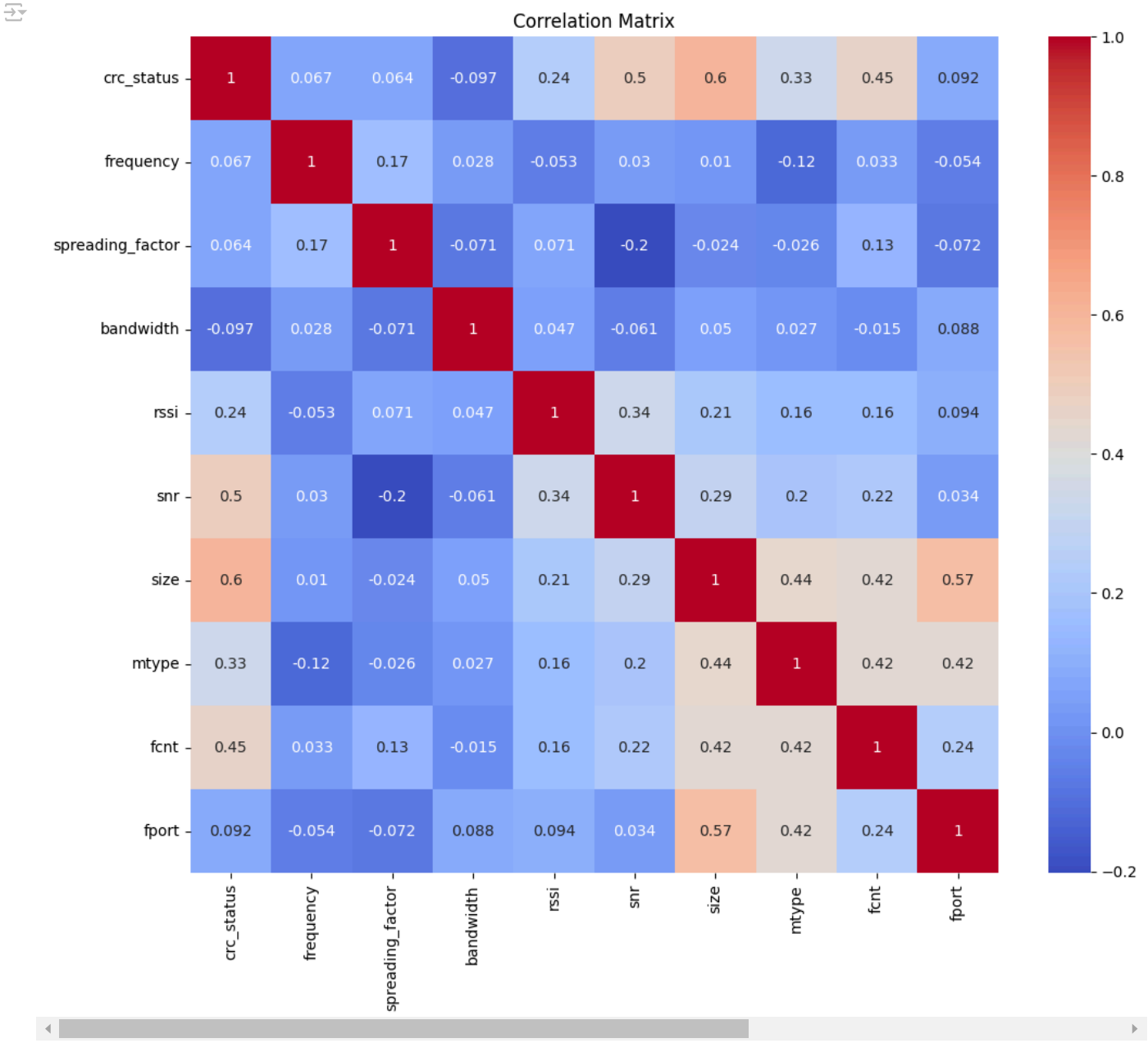

Distribution of mtype
Count
mtype
70000
60000
50000
40000
30000
20000
10000
0
0
20
40
60
80
100


Distribution of fcnt
Count
fcnt
0
5000
10000
15000
20000
25000
30000
35000
40000
0
10000
20000
30000
40000
50000
60000


Distribution of fport
Count
fport
0
10000
20000
30000
40000
50000
60000
70000
0
50
100
150
200
250

```
corr_matrix = numeric_df.corr()

# Plot heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```




```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split



# Specify your features and target
features = ['frequency', 'spreading_factor', 'bandwidth', 'rssi', 'snr', 'size', 'mtype', 'fcnt', 'fport']
target = 'crc_status'

X = df[features]
y = df[target]

# Handle categorical variables
X = pd.get_dummies(X, columns=['mtype'], drop_first=True)

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Create and fit the model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Get feature importances
importances = model.feature_importances_
feature_importances = pd.DataFrame({'feature': X.columns, 'importance': importances})
feature_importances = feature_importances.sort_values(by='importance', ascending=False)

# Select top N features
top_n = 3
best_features = feature_importances.head(top_n)['feature'].tolist()

print("Best features selected:", best_features)

# # Final DataFrame with selected features and target
# X_final = numeric_df[best_features + [target]]
```

```
Best features selected: ['size', 'fport', 'fcnt']
```

```
# Final DataFrame with selected features and target
X_final = numeric_df[best_features + [target]]
X_final
```

	size	fport	fcnt	crc_status
0	29	5.0	8924	1
1	29	5.0	8924	1
2	29	5.0	6895	1
3	29	5.0	6895	1
4	0	-1.0	-1	-1
...	...	...	...	...
108420	42	1.0	4026	1
108421	0	-1.0	-1	-1
108422	47	254.0	63071	-1
108423	71	217.0	14786	-1