


```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import SelectKBest, f_classif, mutual_info_classif
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectKBest, f_classif, mutual_info_classif
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, ConfusionMatrixDisplay
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```


```
df1 = pd.read_csv('/content/ddaefda0913b36051550.csv')
df2 = pd.read_csv('/content/6344ec94bb4449051550.csv')
df3 = pd.read_csv('/content/a04662ab156537051550.csv')
```

```
df1.head() # print first 5 rows - df1.tail()
```



| | latitude | longitude | brightness | scan | track | acq_date | acq_time | satellite | instrument | confidence | version | bright_t31 | frp | daynigt |
|---|----------|-----------|------------|------|-------|------------|----------|-----------|------------|------------|---------|------------|------|---------|
| 0 | 28.0993 | 96.9983 | 303.0 | 1.1 | 1.1 | 2021-01-01 | 409 | Terra | MODIS | 44 | 6.03 | 292.6 | 8.6 | |
| 1 | 30.0420 | 79.6492 | 301.8 | 1.4 | 1.2 | 2021-01-01 | 547 | Terra | MODIS | 37 | 6.03 | 287.4 | 9.0 | |
| 2 | 30.0879 | 78.8579 | 300.2 | 1.3 | 1.1 | 2021-01-01 | 547 | Terra | MODIS | 8 | 6.03 | 286.5 | 5.4 | |
| 3 | 30.0408 | 80.0501 | 302.0 | 1.5 | 1.2 | 2021-01-01 | 547 | Terra | MODIS | 46 | 6.03 | 287.7 | 10.7 | |
| 4 | 30.6565 | 78.9668 | 300.9 | 1.3 | 1.1 | 2021-01-01 | 547 | Terra | MODIS | 43 | 6.03 | 287.6 | 9.0 | |

```
df2.head()
```



| | latitude | longitude | brightness | scan | track | acq_date | acq_time | satellite | instrument | confidence | version | bright_t31 | frp | daynigt |
|---|----------|-----------|------------|------|-------|------------|----------|-----------|------------|------------|---------|------------|------|---------|
| 0 | 30.1138 | 80.0756 | 300.0 | 1.2 | 1.1 | 2022-01-01 | 511 | Terra | MODIS | 7 | 6.03 | 288.4 | 7.1 | |
| 1 | 23.7726 | 86.2078 | 306.1 | 1.6 | 1.2 | 2022-01-01 | 512 | Terra | MODIS | 62 | 6.03 | 293.5 | 10.4 | |
| 2 | 22.2080 | 84.8627 | 304.8 | 1.4 | 1.2 | 2022-01-01 | 512 | Terra | MODIS | 42 | 6.03 | 293.3 | 5.8 | |
| 3 | 23.7621 | 86.3946 | 306.9 | 1.6 | 1.2 | 2022-01-01 | 512 | Terra | MODIS | 38 | 6.03 | 295.2 | 9.3 | |
| 4 | 23.6787 | 86.0891 | 303.6 | 1.5 | 1.2 | 2022-01-01 | 512 | Terra | MODIS | 52 | 6.03 | 293.1 | 7.2 | |

Next steps:


Generate code with df2


 View recommended plots

 New interactive sheet

```
df3.head()
```

 What can I help you build?



| | latitude | longitude | brightness | scan | track | acq_date | acq_time | satellite | instrument | confidence | version | bright_t31 | frp | daynigt |
|---|----------|-----------|------------|------|-------|------------|----------|-----------|------------|------------|---------|------------|------|---------|
| 0 | 9.3280 | 77.6247 | 318.0 | 1.1 | 1.0 | 2023-01-01 | 821 | Aqua | MODIS | 62 | 61.03 | 305.0 | 7.6 | |
| 1 | 10.4797 | 77.9378 | 313.8 | 1.0 | 1.0 | 2023-01-01 | 822 | Aqua | MODIS | 58 | 61.03 | 299.4 | 4.3 | |
| 2 | 13.2478 | 77.2639 | 314.7 | 1.0 | 1.0 | 2023-01-01 | 822 | Aqua | MODIS | 55 | 61.03 | 302.4 | 4.9 | |
| 3 | 12.2994 | 78.4085 | 314.3 | 1.0 | 1.0 | 2023-01-01 | 822 | Aqua | MODIS | 58 | 61.03 | 301.9 | 4.8 | |
| 4 | 14.1723 | 75.5024 | 338.4 | 1.2 | 1.1 | 2023-01-01 | 823 | Aqua | MODIS | 88 | 61.03 | 305.3 | 41.5 | |


Next steps:

[Generate code with df3](#)

[View recommended plots](#)


[New interactive sheet](#)

```
df = pd.concat([df1, df2, df3], ignore_index=True)
df.head()
```




| | latitude | longitude | brightness | scan | track | acq_date | acq_time | satellite | instrument | confidence | version | bright_t31 | frp | daynigt |
|---|----------|-----------|------------|------|-------|------------|----------|-----------|------------|------------|---------|------------|------|---------|
| 0 | 28.0993 | 96.9983 | 303.0 | 1.1 | 1.1 | 2021-01-01 | 409 | Terra | MODIS | 44 | 6.03 | 292.6 | 8.6 | |
| 1 | 30.0420 | 79.6492 | 301.8 | 1.4 | 1.2 | 2021-01-01 | 547 | Terra | MODIS | 37 | 6.03 | 287.4 | 9.0 | |
| 2 | 30.0879 | 78.8579 | 300.2 | 1.3 | 1.1 | 2021-01-01 | 547 | Terra | MODIS | 8 | 6.03 | 286.5 | 5.4 | |
| 3 | 30.0408 | 80.0501 | 302.0 | 1.5 | 1.2 | 2021-01-01 | 547 | Terra | MODIS | 46 | 6.03 | 287.7 | 10.7 | |
| 4 | 30.6565 | 78.9668 | 300.9 | 1.3 | 1.1 | 2021-01-01 | 547 | Terra | MODIS | 43 | 6.03 | 287.6 | 9.0 | |

```
df.shape # rows and cols
```




```
(271217, 15)
```

```
df.info() # dt, memc
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271217 entries, 0 to 271216
Data columns (total 15 columns):
#   Column      Non-Null Count  Dtype
---  -
0   latitude    271217 non-null float64
1   longitude   271217 non-null float64
2   brightness  271217 non-null float64
3   scan        271217 non-null float64
4   track       271217 non-null float64
5   acq_date    271217 non-null object
6   acq_time    271217 non-null int64
7   satellite   271217 non-null object
8   instrument  271217 non-null object
9   confidence  271217 non-null int64
10  version     271217 non-null float64
11  bright_t31  271217 non-null float64
12  frp         271217 non-null float64
13  daynight    271217 non-null object
14  type        271217 non-null int64
dtypes: float64(8), int64(3), object(4)
memory usage: 31.0+ MB
```


```
# Any missing values?
df.isnull().sum()
```



| | |
|------------|---|
| | 0 |
| latitude | 0 |
| longitude | 0 |
| brightness | 0 |
| scan | 0 |
| track | 0 |
| acq_date | 0 |
| acq_time | 0 |
| satellite | 0 |
| instrument | 0 |
| confidence | 0 |
| version | 0 |
| bright_t31 | 0 |
| frp | 0 |
| daynight | 0 |
| type | 0 |


dtype: int64

```
df.duplicated().sum()
```



| |
|-------------|
| np.int64(0) |
|-------------|

```
# List out column names to check
df.columns
```



| |
|--|
| Index(['latitude', 'longitude', 'brightness', 'scan', 'track', 'acq_date', 'acq_time', 'satellite', 'instrument', 'confidence', 'version', 'bright_t31', 'frp', 'daynight', 'type'], dtype='object') |
|--|

```
df.describe().T # statistics of dataset - numbers!
```



| | count | mean | std | min | 25% | 50% | 75% | max |
|------------|----------|------------|------------|----------|----------|----------|----------|-----------|
| latitude | 271217.0 | 23.947505 | 4.919846 | 8.1362 | 20.9655 | 23.7888 | 27.7827 | 34.9734 |
| longitude | 271217.0 | 81.284024 | 6.559071 | 68.4526 | 75.8802 | 79.3209 | 84.7559 | 97.1044 |
| brightness | 271217.0 | 323.719192 | 14.147221 | 300.0000 | 314.5000 | 322.0000 | 330.7000 | 505.7000 |
| scan | 271217.0 | 1.421732 | 0.630742 | 1.0000 | 1.0000 | 1.2000 | 1.5000 | 4.8000 |
| track | 271217.0 | 1.152716 | 0.201943 | 1.0000 | 1.0000 | 1.1000 | 1.2000 | 2.0000 |
| acq_time | 271217.0 | 824.623755 | 353.966965 | 321.0000 | 648.0000 | 756.0000 | 825.0000 | 2202.0000 |
| confidence | 271217.0 | 64.065081 | 18.165329 | 0.0000 | 54.0000 | 66.0000 | 76.0000 | 100.0000 |
| version | 271217.0 | 21.933778 | 24.935515 | 6.0300 | 6.0300 | 6.0300 | 61.0300 | 61.0300 |
| bright_t31 | 271217.0 | 303.499177 | 8.282440 | 267.2000 | 298.2000 | 302.5000 | 309.2000 | 400.1000 |
| frp | 271217.0 | 27.722058 | 81.017471 | 0.0000 | 8.7000 | 13.5000 | 24.5000 | 6961.8000 |
| type | 271217.0 | 0.100385 | 0.437215 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 3.0000 |

```
# Check Unique values of target variable
df.type.value_counts()
```

```

↗
count
type
0    257625
2     13550
3         42

dtype: int64

```

```

# Check unique and n unique for all categorical features
for col in df.columns:
    if df[col].dtype == 'object':
        print(f"Column: {col}")
        print(f"Unique values: {df[col].unique()}")
        print(f"Number of unique values: {df[col].nunique()}")
        print("-" * 50)

```

```

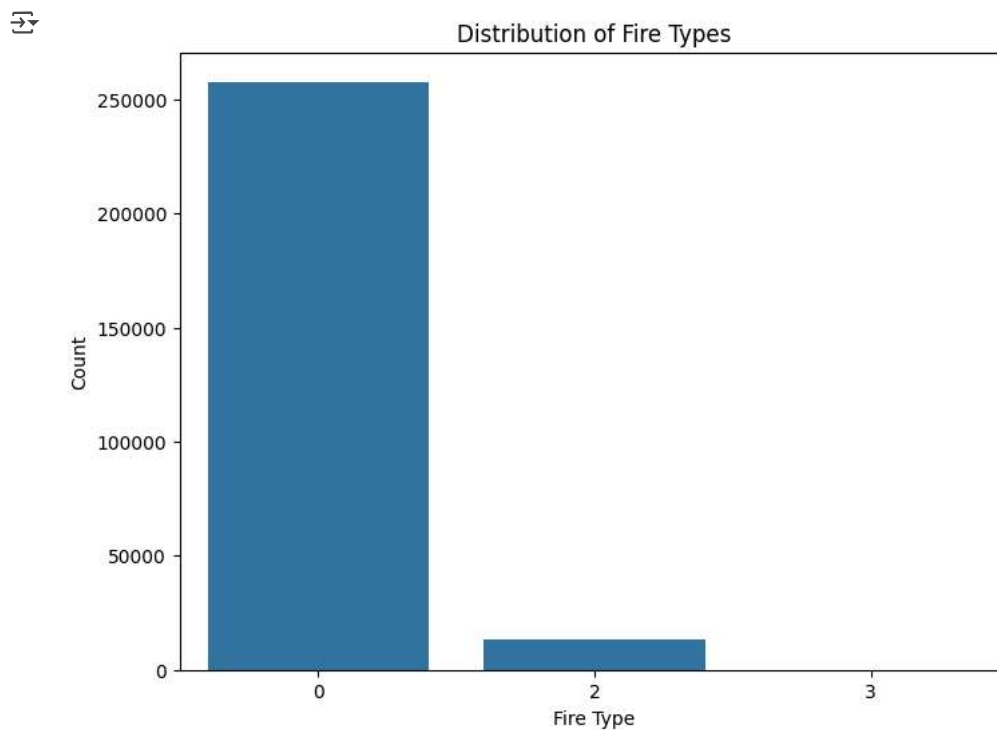
↗
Column: acq_date
Unique values: ['2021-01-01' '2021-01-02' '2021-01-03' ... '2023-12-29' '2023-12-30'
                '2023-12-31']
Number of unique values: 1088
-----
Column: satellite
Unique values: ['Terra' 'Aqua']
Number of unique values: 2
-----
Column: instrument
Unique values: ['MODIS']
Number of unique values: 1
-----
Column: daynight
Unique values: ['D' 'N']
Number of unique values: 2
-----

```

```

# Count plot for 'type'
plt.figure(figsize=(8, 6))
sns.countplot(x='type', data=df)
plt.title('Distribution of Fire Types')
plt.xlabel('Fire Type')
plt.ylabel('Count')
plt.show()

```

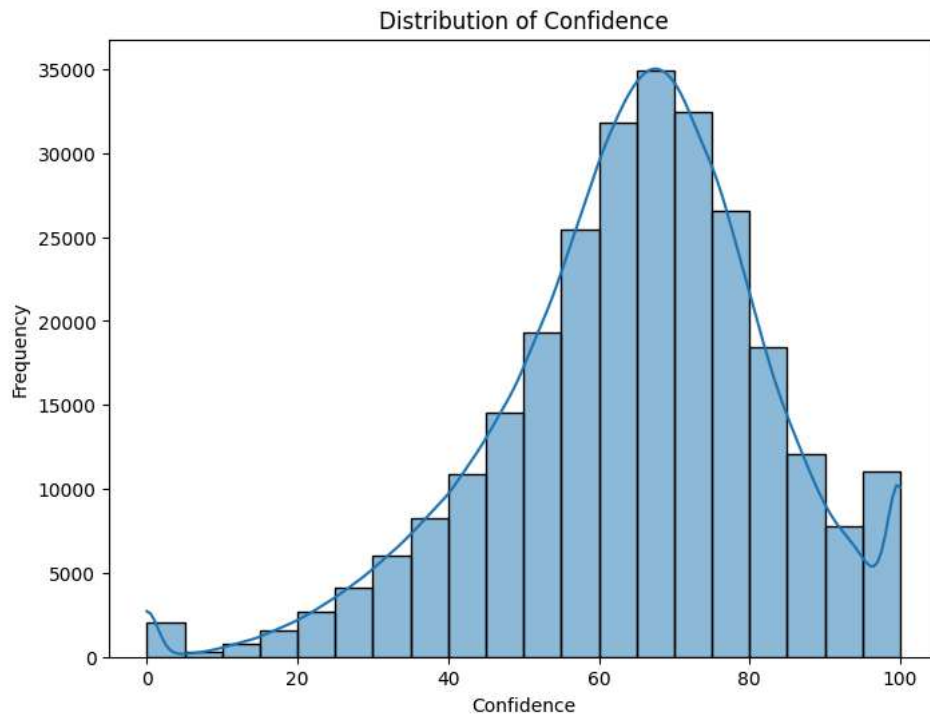


```

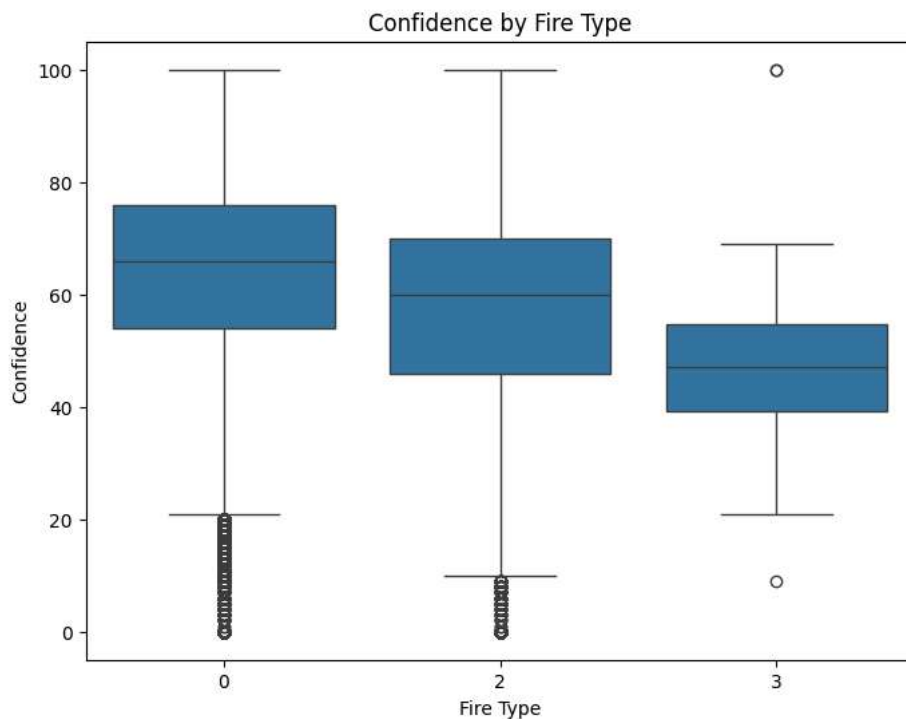
# Histogram of 'confidence'

```

```
# Histogram of Confidence
plt.figure(figsize=(8, 6))
sns.histplot(df['confidence'], bins=20, kde=True)
plt.title('Distribution of Confidence')
plt.xlabel('Confidence')
plt.ylabel('Frequency')
plt.show()
```

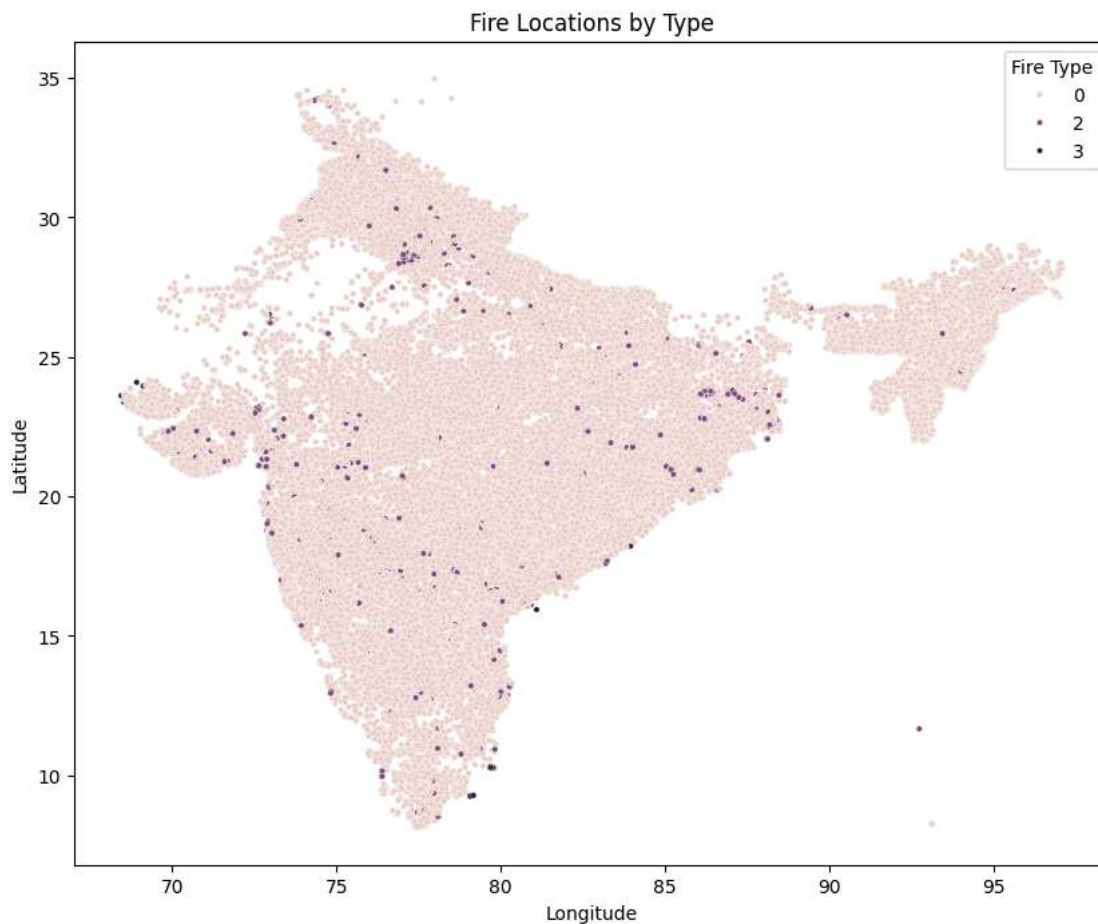


```
# Box plot for 'confidence' by 'type'
plt.figure(figsize=(8, 6))
sns.boxplot(x='type', y='confidence', data=df)
plt.title('Confidence by Fire Type')
plt.xlabel('Fire Type')
plt.ylabel('Confidence')
plt.show()
```

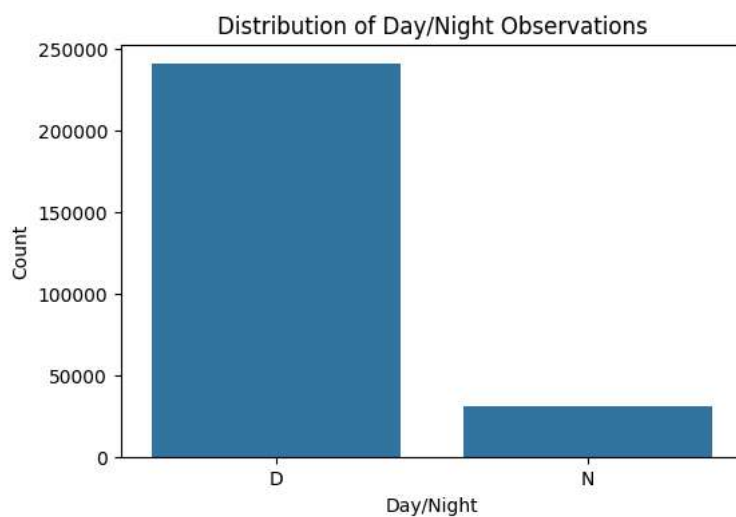


```
# Scatter plot of 'latitude' vs 'longitude'
```

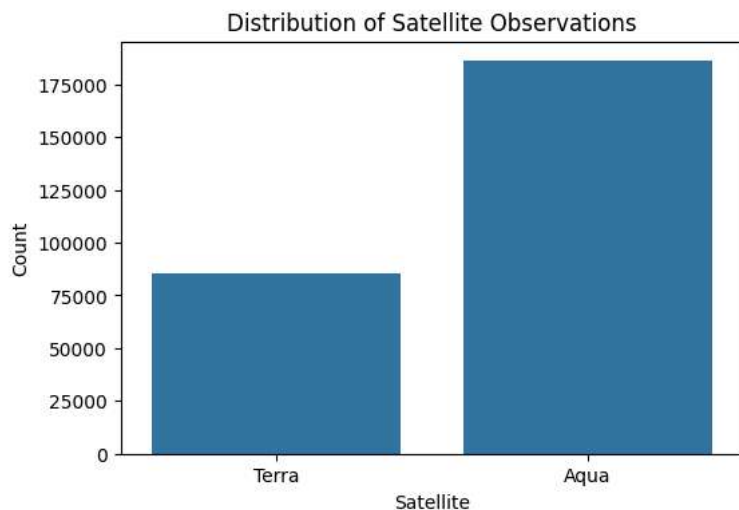
```
plt.figure(figsize=(10, 8))
sns.scatterplot(x='longitude', y='latitude', data=df, hue='type', s=10)
plt.title('Fire Locations by Type')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.legend(title='Fire Type')
plt.show()
```



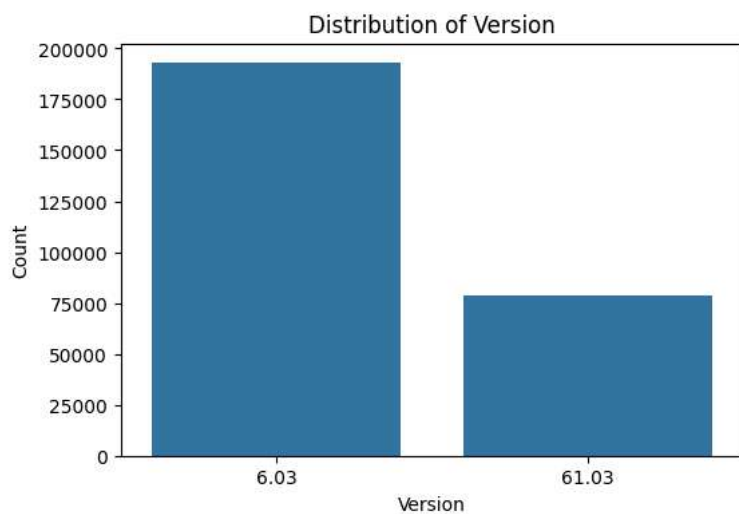
```
# Count plot for 'daynight'
plt.figure(figsize=(6, 4))
sns.countplot(x='daynight', data=df)
plt.title('Distribution of Day/Night Observations')
plt.xlabel('Day/Night')
plt.ylabel('Count')
plt.show()
```



```
# Count plot for 'Satellite'
plt.figure(figsize=(6, 4))
sns.countplot(x='satellite', data=df)
plt.title('Distribution of Satellite Observations')
plt.xlabel('Satellite')
plt.ylabel('Count')
plt.show()
```

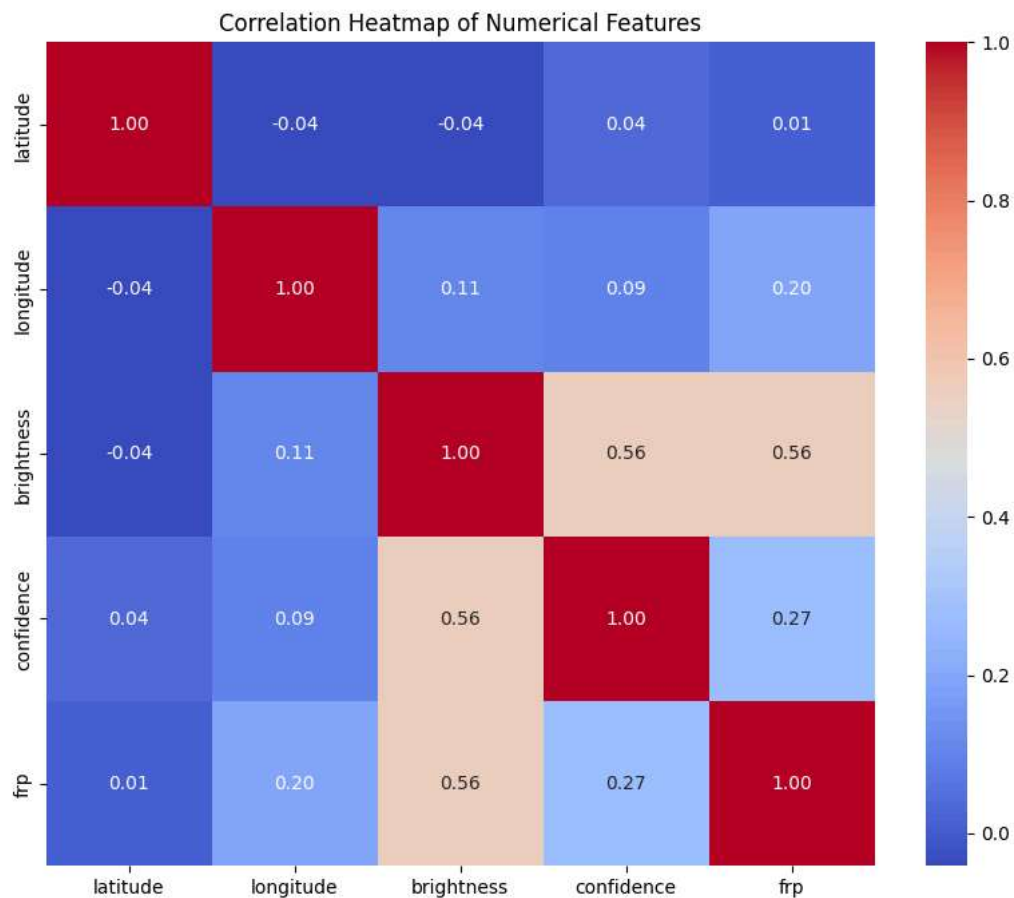


```
# Count plot for 'version'
plt.figure(figsize=(6, 4))
sns.countplot(x='version', data=df)
plt.title('Distribution of Version')
plt.xlabel('Version')
plt.ylabel('Count')
plt.show()
```



```
#this code take more time
#Pairplot for numerical features (subset)
#sns.pairplot(df[['latitude', 'longitude', 'brightness', 'confidence', 'frp', 'type']], hue='type', diag_kind='kde')
#plt.suptitle('Pairplot of Numerical Features')
#plt.show()
```

```
# Heatmap of correlations between numerical features
plt.figure(figsize=(10, 8))
correlation_matrix = df[['latitude', 'longitude', 'brightness', 'confidence', 'frp']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Features')
plt.show()
```



```
numerical_cols = df.select_dtypes(include=np.number).columns
```

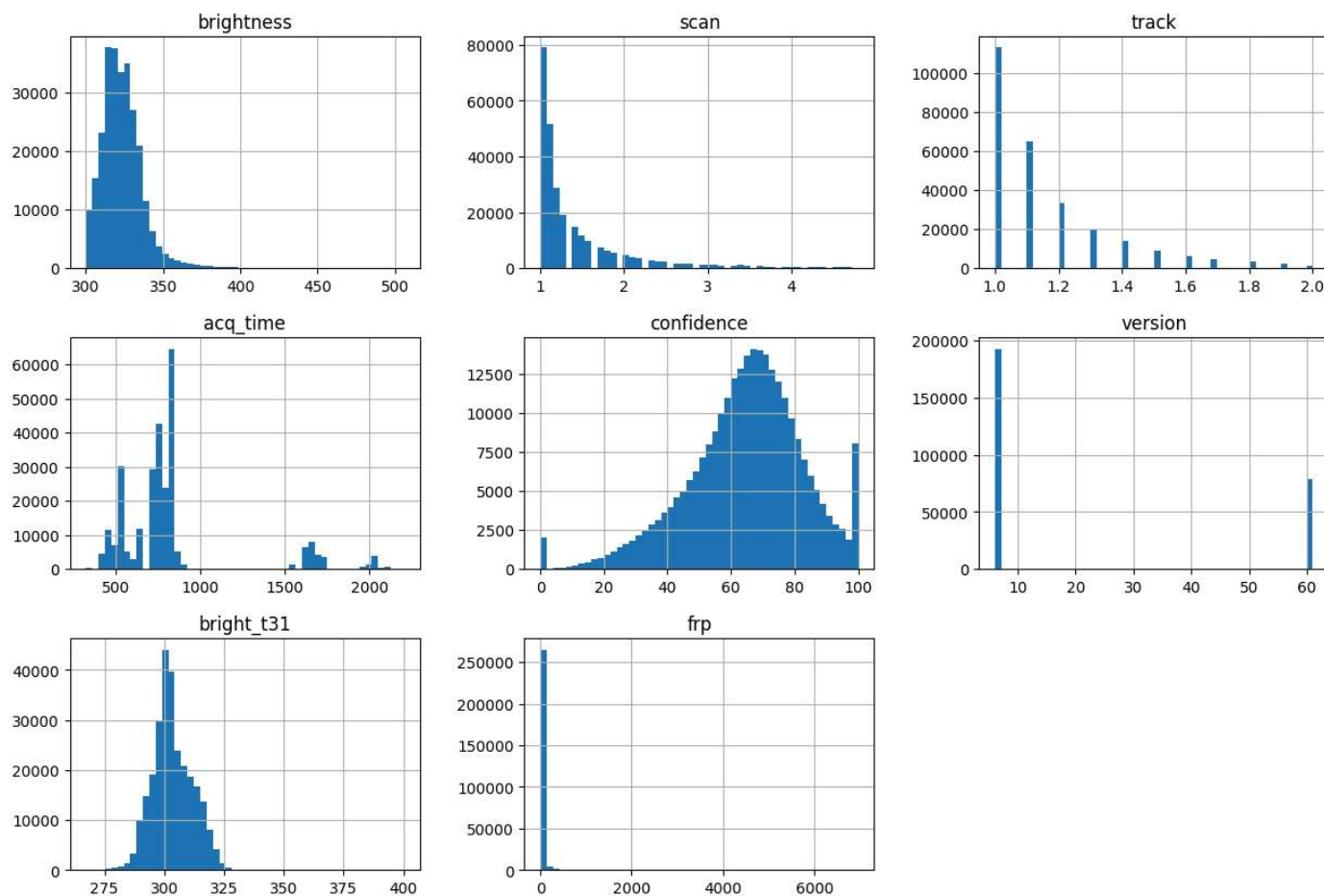
```
numerical_cols
```

```
Index(['latitude', 'longitude', 'brightness', 'scan', 'track', 'acq_time',  
      'confidence', 'version', 'bright_t31', 'frp', 'type'],  
      dtype='object')
```

```
numerical_cols = ['brightness', 'scan', 'track', 'acq_time', 'confidence', 'version', 'bright_t31', 'frp']  
df[numerical_cols].hist(bins=50, figsize=(15, 10))  
plt.suptitle('Histograms of Numerical Features')  
plt.show()
```




Histograms of Numerical Features



```
import statsmodels.api as sm
import scipy.stats as stats

# List of numerical features to check for distribution
numerical_features = ['brightness', 'confidence', 'frp', 'bright_t31', 'scan', 'track']

for feature in numerical_features:
    print(f"Analyzing distribution for: {feature}")

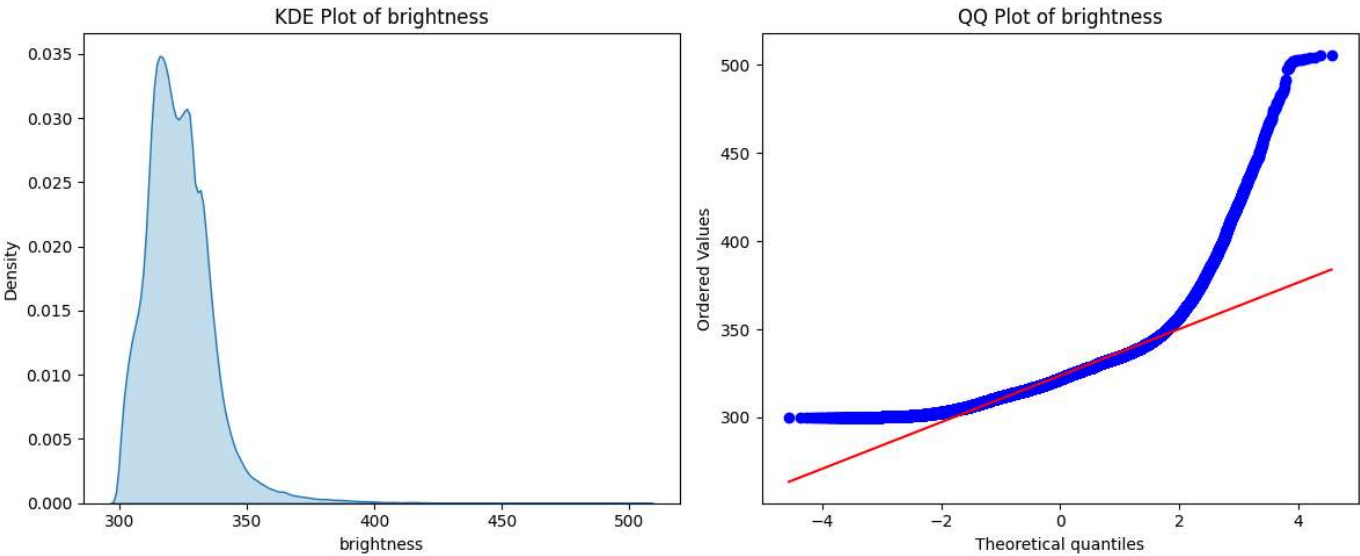
    # KDE Plot
    plt.figure(figsize=(12, 5))

    plt.subplot(1, 2, 1)
    sns.kdeplot(df[feature], fill=True)
    plt.title(f'KDE Plot of {feature}')
    plt.xlabel(feature)
    plt.ylabel('Density')

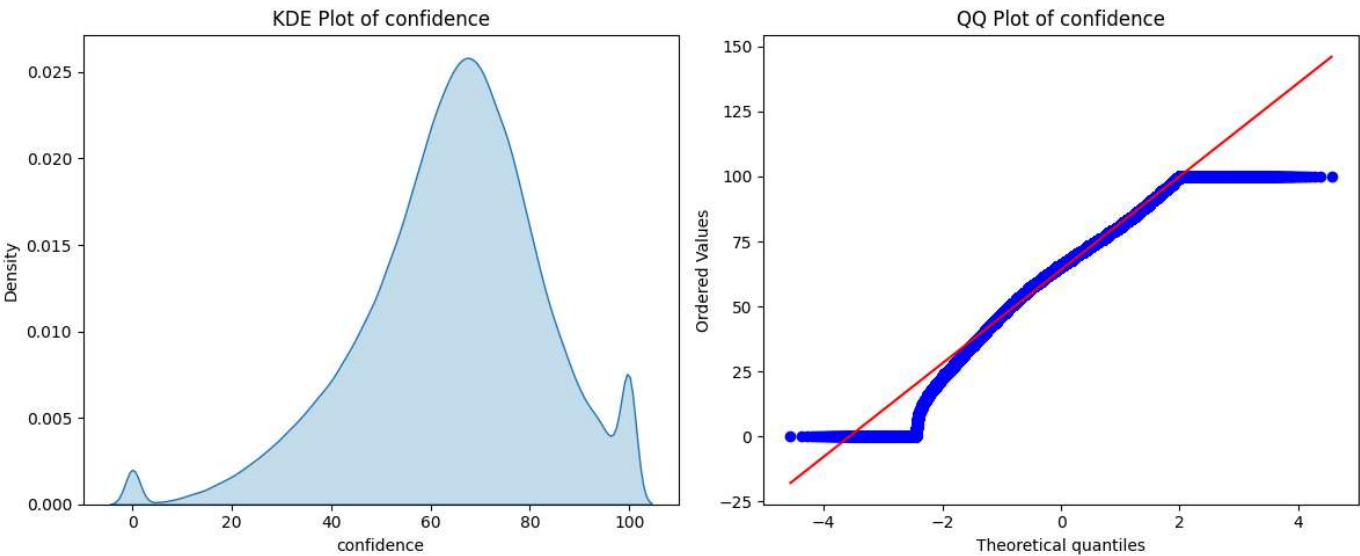
    # QQ Plot
    plt.subplot(1, 2, 2)
    stats.probplot(df[feature], dist="norm", plot=plt)
    plt.title(f'QQ Plot of {feature}')

plt.tight_layout()
plt.show()
print("-" * 50)
```

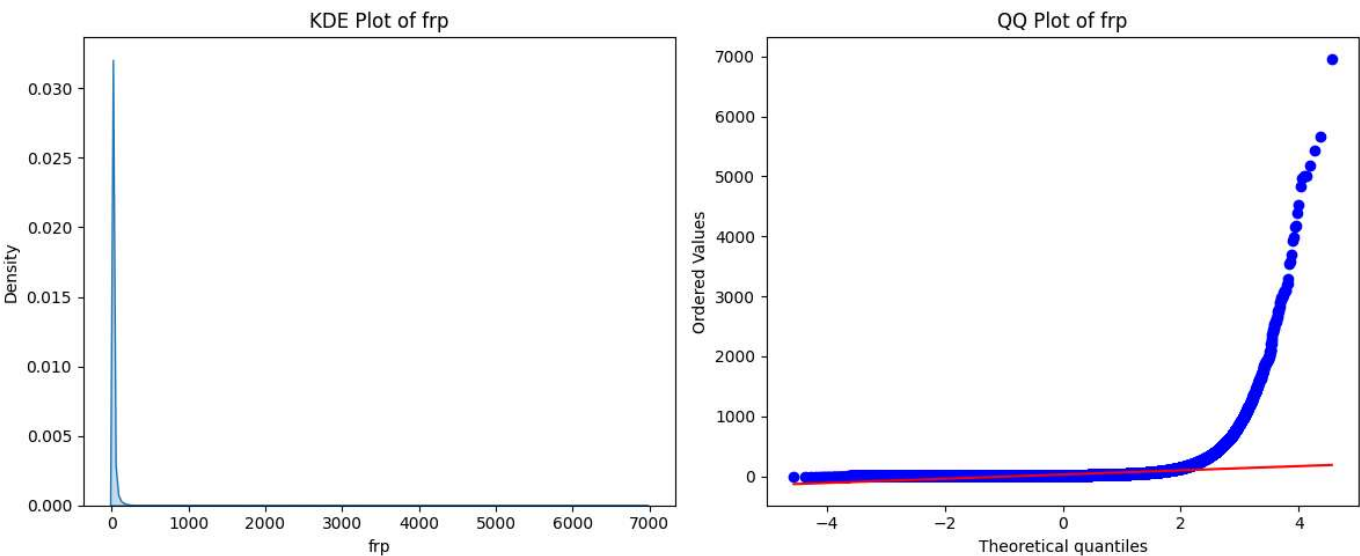
Analyzing distribution for: brightness



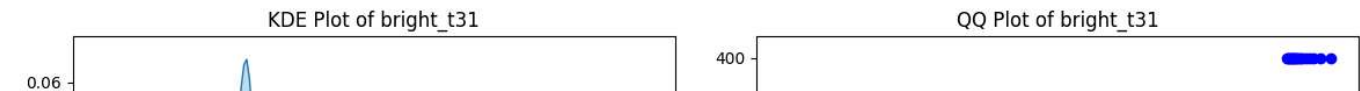
Analyzing distribution for: confidence

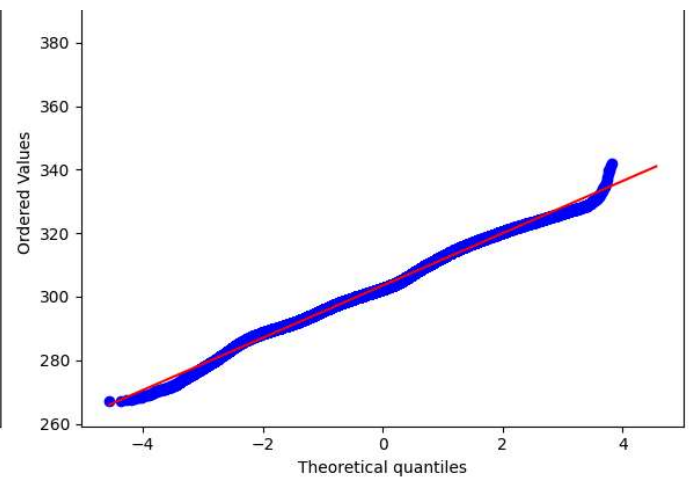
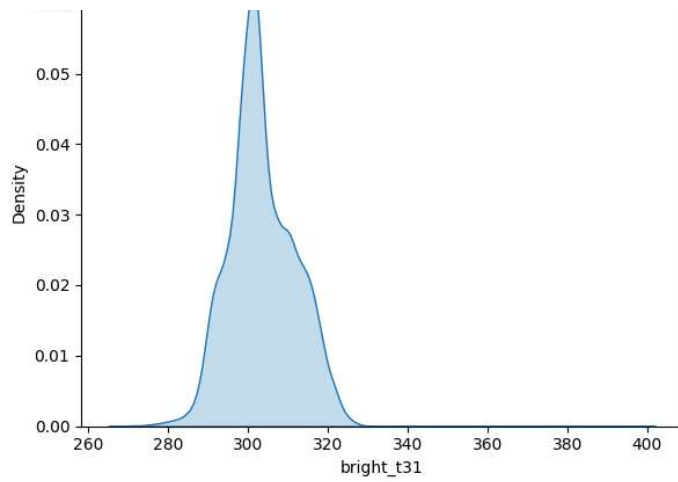


Analyzing distribution for: frp

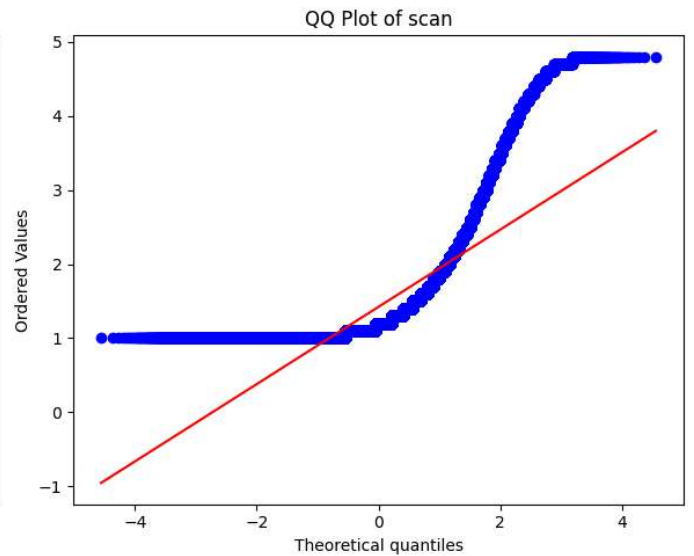
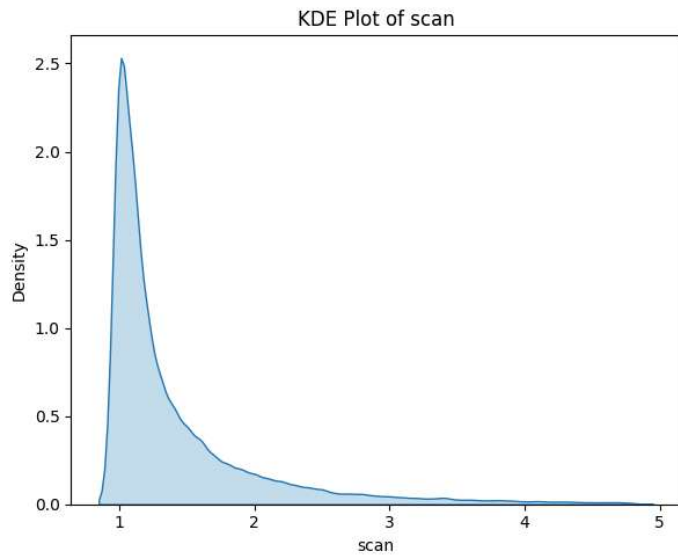


Analyzing distribution for: bright_t31

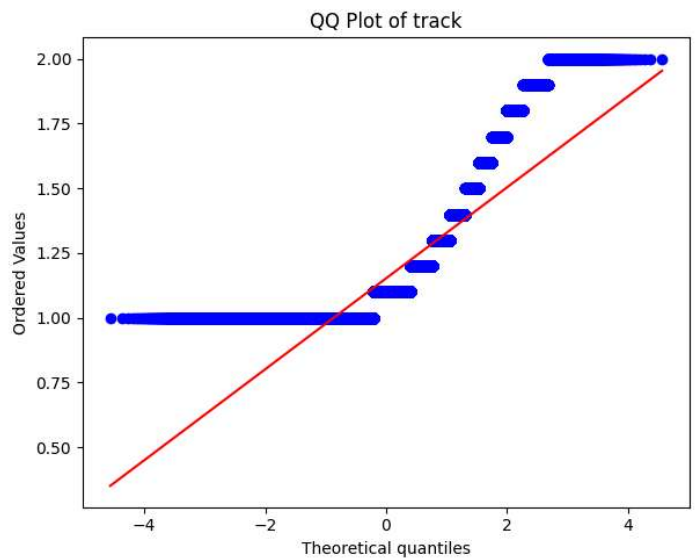
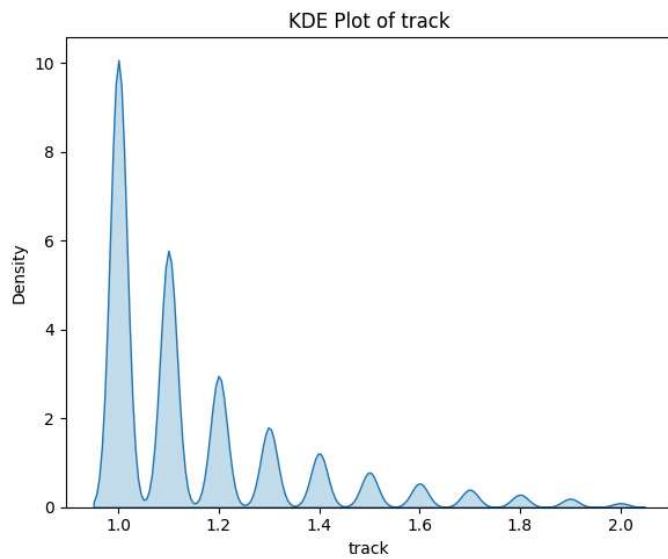




Analyzing distribution for: scan




Analyzing distribution for: track




```
# --- Temporal Analysis ---
# Convert 'acq_date' to datetime objects
df['acq_date'] = pd.to_datetime(df['acq_date'])
# Extract temporal features
df['year'] = df['acq_date'].dt.year
df['month'] = df['acq_date'].dt.month
df['day_of_week'] = df['acq_date'].dt.dayofweek # Monday=0, Sunday=6
df['day_of_year'] = df['acq_date'].dt.dayofyear
df['hour'] = df['acq_time'].astype(str).str[:2].astype(int) # Assuming acq_time is HHMM
```

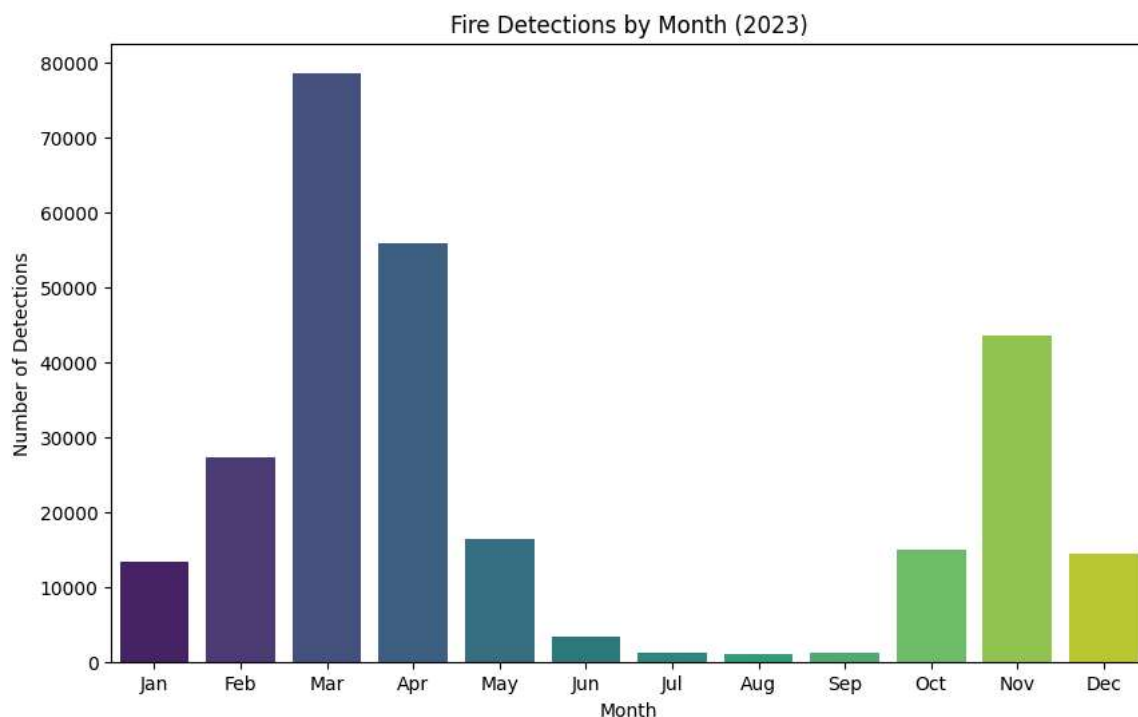
Double-click (or enter) to edit

```
# Visualize fire detections over months
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='month', palette='viridis')
plt.title('Fire Detections by Month (2023)')
plt.xlabel('Month')
plt.ylabel('Number of Detections')
plt.xticks(ticks=range(12), labels=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
plt.show()
```


 /tmp/ipython-input-31-3766763484.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend`

```
sns.countplot(data=df, x='month', palette='viridis')
```

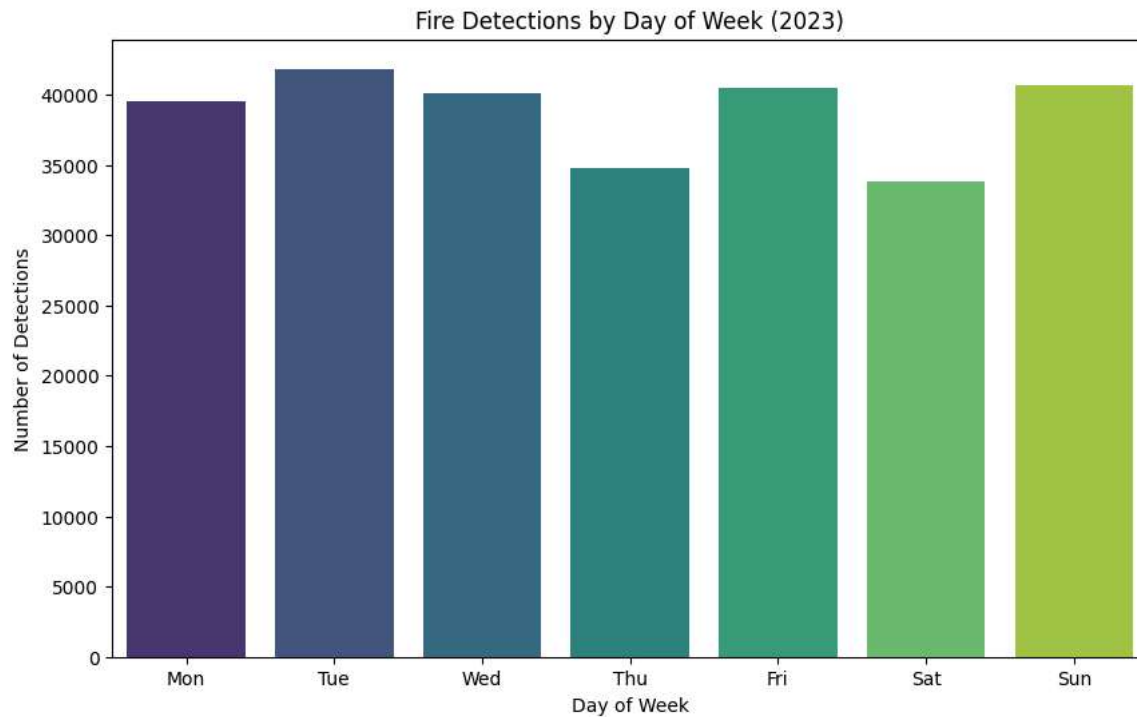


```
# Visualize fire detections by day of the week
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='day_of_week', palette='viridis')
plt.title('Fire Detections by Day of Week (2023)')
plt.xlabel('Day of Week')
plt.ylabel('Number of Detections')
plt.xticks(ticks=range(7), labels=['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'])
plt.show()
```

 /tmp/ipython-input-32-714612371.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend`

```
sns.countplot(data=df, x='day_of_week', palette='viridis')
```



```
# Visualize outliers using box plots for key numerical features
plt.figure(figsize=(12, 8))
sns.boxplot(data=df[numerical_cols])
plt.title('Box Plots for Key Numerical Features')
plt.ylabel('Value')
plt.show()
```