

```
In [1]: import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

```
In [3]: df = pd.read_csv('bank.csv')
```

```
In [7]: df.head(10)
```

```
Out[7]:
```

	age	job	marital	education	default	balance	housing	loan	contact	day
0	58	management	married	tertiary	no	2143	yes	no	unknown	5
1	44	technician	single	secondary	no	29	yes	no	unknown	5
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5
4	33	unknown	single	unknown	no	1	no	no	unknown	5
5	35	management	married	tertiary	no	231	yes	no	unknown	5
6	28	management	single	tertiary	no	447	yes	yes	unknown	5
7	42	entrepreneur	divorced	tertiary	yes	2	yes	no	unknown	5
8	58	retired	married	primary	no	121	yes	no	unknown	5
9	43	technician	single	secondary	no	593	yes	no	unknown	5

```
In [9]: df.dtypes
```

```
Out[9]: age          int64
job            object
marital        object
education      object
default        object
balance        int64
housing        object
loan           object
contact        object
day            int64
month          object
duration       int64
campaign       int64
pdays         int64
previous       int64
poutcome      object
deposit        object
dtype: object
```

```
In [11]: df.isnull().sum()
```

```
Out[11]: age      0
        job      0
        marital   0
        education 0
        default   0
        balance   0
        housing   0
        loan      0
        contact   0
        day       0
        month     0
        duration  0
        campaign  0
        pdays    0
        previous  0
        poutcome  0
        deposit   0
        dtype: int64
```

```
In [13]: df.replace("unknown", np.nan, inplace=True)
```

```
In [15]: df.isnull().sum()
```

```
Out[15]: age      0
        job      288
        marital   0
        education 1857
        default   0
        balance   0
        housing   0
        loan      0
        contact   13020
        day       0
        month     0
        duration  0
        campaign  0
        pdays    0
        previous  0
        poutcome  36959
        deposit   0
        dtype: int64
```

```
In [19]: #Deleting columns that have 30% missing values
missing_percentage = df.isnull().mean() * 100
threshold = 30

columns_to_drop = missing_percentage[missing_percentage >
threshold].index

df = df.drop(columns=columns_to_drop)

print(f"Columns dropped: {list(columns_to_drop)}")
print(f"Remaining columns: {df.columns}")
```

```
Columns dropped: ['poutcome']
Remaining columns: Index(['age', 'job', 'marital', 'education', 'default', 'balance', 'housing',
                        'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays',
                        'previous', 'deposit'],
                        dtype='object')
```

```
In [23]: for column in df.select_dtypes(include=['number']).columns:
        mean_value = df[column].mean()
        df[column] = df[column].fillna(mean_value)
        for column in df.select_dtypes(include=['object']).columns:
            mode_value = df[column].mode()[0]
            df[column] = df[column].fillna(mode_value)
        df.isnull().sum()
```

```
Out[23]: age          0
        job          0
        marital      0
        education    0
        default      0
        balance      0
        housing      0
        loan         0
        contact      0
        day          0
        month        0
        duration     0
        campaign     0
        pdays        0
        previous     0
        deposit      0
        dtype: int64
```

```
In [29]: df.head(10)
```

```
Out[29]:
```

	age	job	marital	education	default	balance	housing	loan	contact	day
0	58	management	married	tertiary	no	2143	yes	no	cellular	5
1	44	technician	single	secondary	no	29	yes	no	cellular	5
2	33	entrepreneur	married	secondary	no	2	yes	yes	cellular	5
3	47	blue-collar	married	secondary	no	1506	yes	no	cellular	5
4	33	blue-collar	single	secondary	no	1	no	no	cellular	5
5	35	management	married	tertiary	no	231	yes	no	cellular	5
6	28	management	single	tertiary	no	447	yes	yes	cellular	5
7	42	entrepreneur	divorced	tertiary	yes	2	yes	no	cellular	5
8	58	retired	married	primary	no	121	yes	no	cellular	5
9	43	technician	single	secondary	no	593	yes	no	cellular	5

```
In [35]: from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    df[column] = label_encoder.fit_transform(df[column])
df.head()
```

```
Out[35]:
```

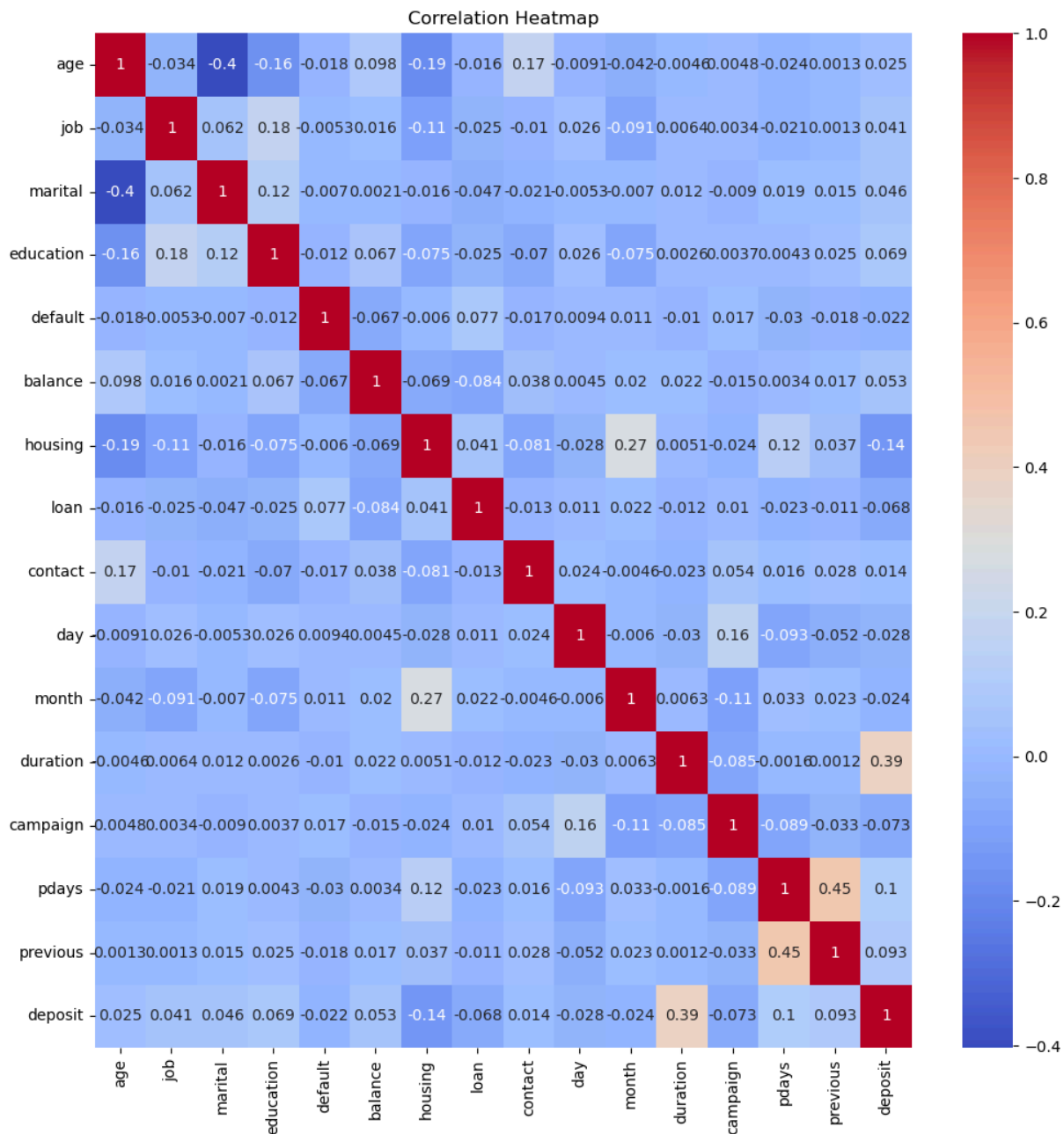
	age	job	marital	education	default	balance	housing	loan	contact	day	month	d
0	58	4	1	2	0	2143	1	0	0	5	8	
1	44	9	2	1	0	29	1	0	0	5	8	
2	33	2	1	1	0	2	1	1	0	5	8	
3	47	1	1	1	0	1506	1	0	0	5	8	
4	33	1	2	1	0	1	0	0	0	5	8	



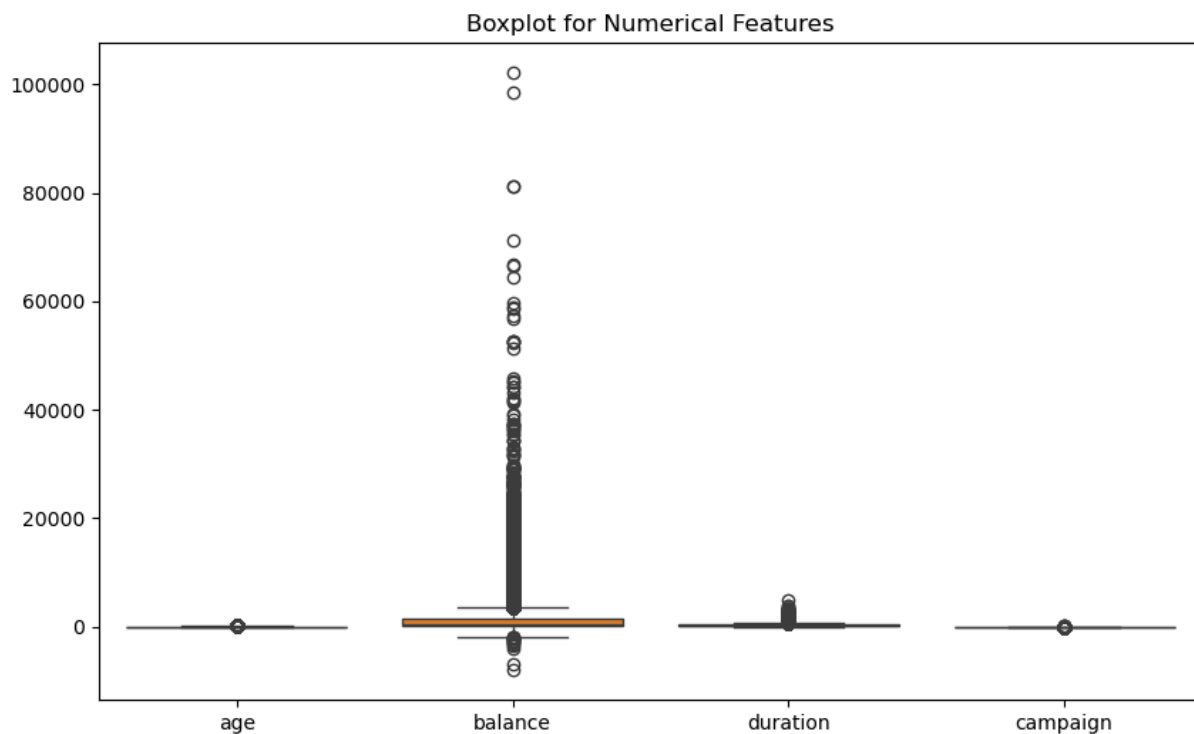
```
In [37]: df.dtypes
```

```
Out[37]: age          int64
job            int32
marital        int32
education      int32
default        int32
balance        int64
housing        int32
loan           int32
contact        int32
day            int64
month          int32
duration       int64
campaign       int64
pdays         int64
previous       int64
deposit        int32
dtype: object
```

```
In [39]: plt.figure(figsize=(12, 12))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



```
In [55]: plt.figure(figsize=(10,6))
sns.boxplot(data=df[['age','balance','duration','campaign']])
plt.title("Boxplot for Numerical Features")
plt.show()
```



```
In [57]: Q1 = df.quantile(0.25)
         Q3 = df.quantile(0.75)

         IQR = Q3 - Q1
```

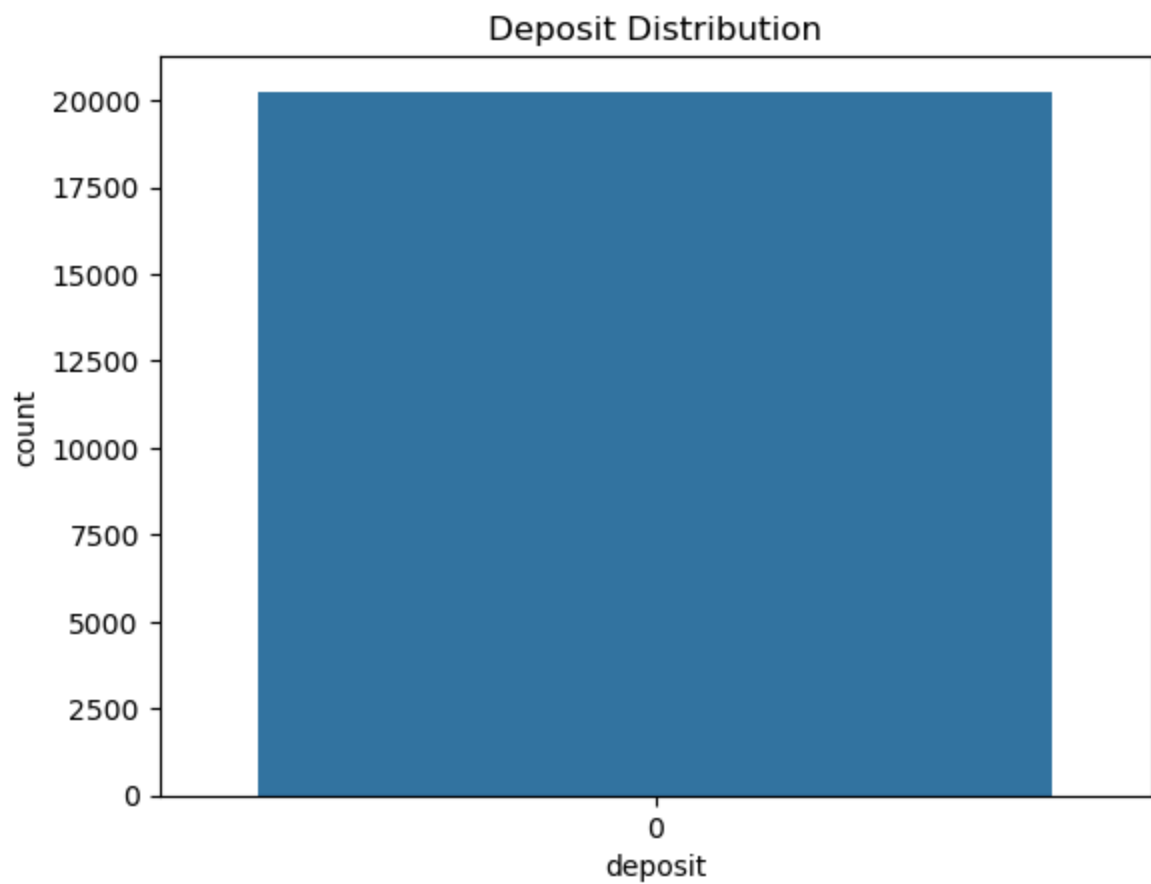
```
In [59]: lower = Q1 - 1.5 * IQR
         upper = Q3 + 1.5 * IQR
```

```
In [61]: df_clean = df[~((df < lower) | (df > upper)).any(axis=1)]
```

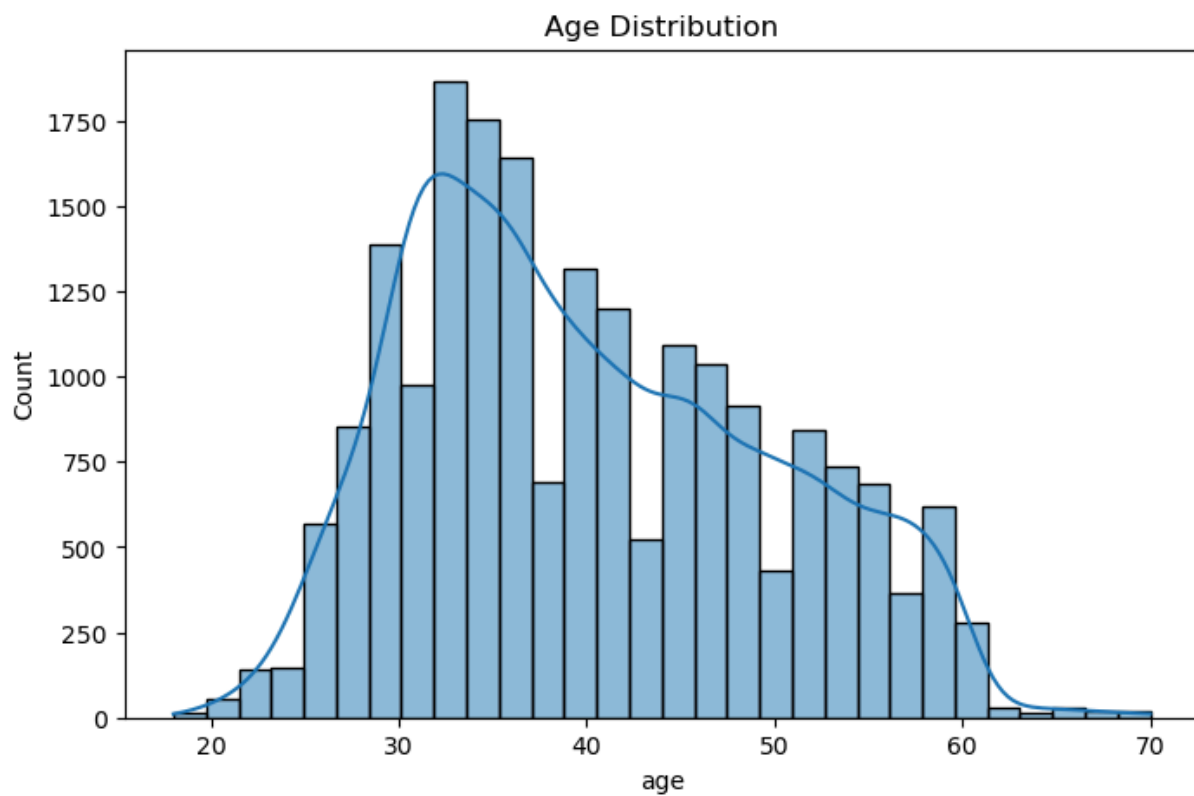
```
In [63]: df_clean.shape
```

```
Out[63]: (20253, 16)
```

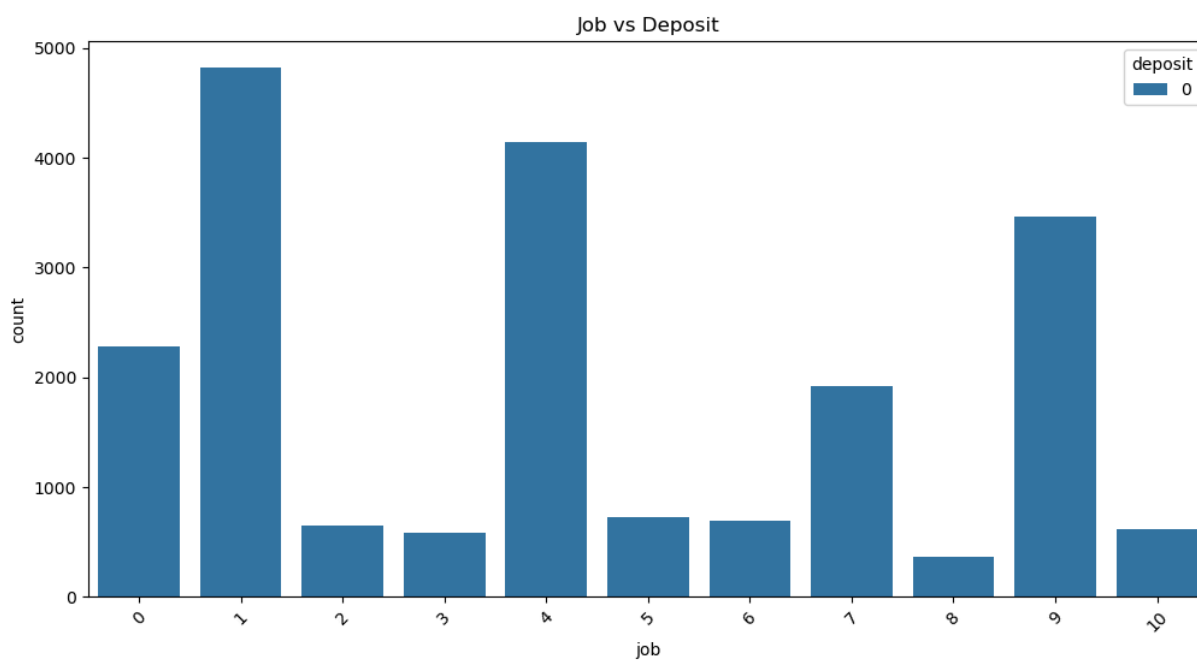
```
In [65]: sns.countplot(x='deposit', data=df_clean)
         plt.title("Deposit Distribution")
         plt.show()
```



```
In [67]: plt.figure(figsize=(8,5))
sns.histplot(df_clean['age'], bins=30, kde=True)
plt.title("Age Distribution")
plt.show()
```



```
In [69]: plt.figure(figsize=(12,6))
sns.countplot(x='job', hue='deposit', data=df_clean)
plt.xticks(rotation=45)
plt.title("Job vs Deposit")
plt.show()
```



```
In [73]: df_clean.head()
```

Out[73]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	d
0	58	4	1	2	0	2143	1	0	0	5	8	
1	44	9	2	1	0	29	1	0	0	5	8	
3	47	1	1	1	0	1506	1	0	0	5	8	
4	33	1	2	1	0	1	0	0	0	5	8	
5	35	4	1	2	0	231	1	0	0	5	8	



In [75]: `df_clean.to_csv("bank_cleaned.csv", index=False)`

In []: