

## Import Libraries

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

## Data Preview

```
df = pd.read_csv("kidney_disease.csv")
df.head()
```

|   | age  | bp   | sg    | al  | su  | rbc    | pc       | pcc        | ba         | bgr   | ... | pcv | wc   | rc  | htn | dm  | cad | appet | pe  | ane |
|---|------|------|-------|-----|-----|--------|----------|------------|------------|-------|-----|-----|------|-----|-----|-----|-----|-------|-----|-----|
| 0 | 48.0 | 80.0 | 1.020 | 1.0 | 0.0 | NaN    | normal   | notpresent | notpresent | 121.0 | ... | 44  | 7800 | 5.2 | yes | yes | no  | good  | no  | no  |
| 1 | 7.0  | 50.0 | 1.020 | 4.0 | 0.0 | NaN    | normal   | notpresent | notpresent | NaN   | ... | 38  | 6000 | NaN | no  | no  | no  | good  | no  | no  |
| 2 | 62.0 | 80.0 | 1.010 | 2.0 | 3.0 | normal | normal   | notpresent | notpresent | 423.0 | ... | 31  | 7500 | NaN | no  | yes | no  | poor  | no  | yes |
| 3 | 48.0 | 70.0 | 1.005 | 4.0 | 0.0 | normal | abnormal | present    | notpresent | 117.0 | ... | 32  | 6700 | 3.9 | yes | no  | no  | poor  | yes | yes |
| 4 | 51.0 | 80.0 | 1.010 | 2.0 | 0.0 | normal | normal   | notpresent | notpresent | 106.0 | ... | 35  | 7300 | 4.6 | no  | no  | no  | good  | no  | no  |

5 rows × 25 columns

## Part 1: Handle Missing Values

### Check Missing Values

```
print("Missing values before handling:\n", df.isnull().sum())
```

```
Missing values before handling:
age          9
bp           12
sg           46
al           45
su           48
rbc          151
pc           64
pcc           4
ba            4
bgr          44
bu           19
sc           17
sod           87
pot           88
hemo         51
pcv          69
wc           105
rc           129
htn           2
dm            2
cad           2
appet         1
pe            1
ane           1
classification 0
dtype: int64
```

## Strategy 1 (Using median)

```
# ✅ Handle missing values
for col in df.select_dtypes(include=['float64', 'int64']).columns:
    df[col] = df[col].fillna(df[col].median())
```

## Strategy 2 (Using mode)

```
# ✅ Handle categorical columns
for col in df.select_dtypes(include=['object']).columns:
    mode_value = df[col].mode()[0]
```

```
df[col] = df[col].fillna(mode_value)
```

```
print("\nMissing values after handling:\n", df.isnull().sum())
```

```
Missing values after handling:
age          0
bp           0
sg           0
al           0
su           0
rbc          0
pc           0
pcc          0
ba           0
bgr          0
bu           0
sc           0
sod          0
pot          0
hemo         0
pcv          0
wc           0
rc           0
htn          0
dm           0
cad          0
appet        0
pe           0
ane         0
classification 0
dtype: int64
```

Result: There is no missing values in this dataset.

## Part 2: Handle Outliers

```
def remove_outliers_iqr(data, columns):
    for col in columns:
        Q1 = data[col].quantile(0.25)
        Q3 = data[col].quantile(0.75)
        IQR = Q3 - Q1
        lower = Q1 - 1.5 * IQR
        upper = Q3 + 1.5 * IQR
        data = data[(data[col] >= lower) & (data[col] <= upper)]
    return data

numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
df_clean = remove_outliers_iqr(df, numeric_cols)

print(f"Shape before removing outliers: {df.shape}")
print(f"Shape after removing outliers: {df_clean.shape}")
```

```
Shape before removing outliers: (398, 25)
Shape after removing outliers: (203, 25)
```

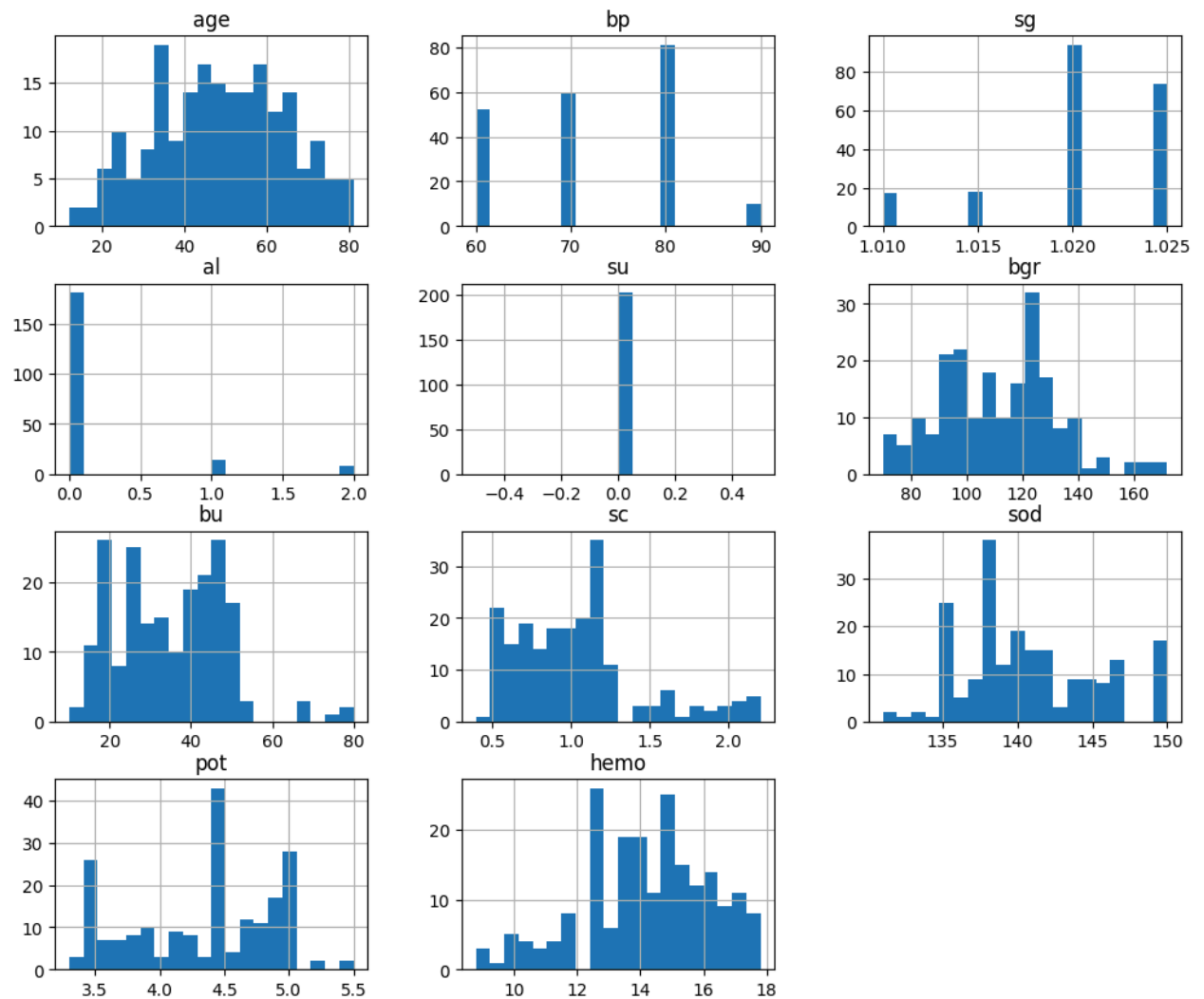
Result: There is no remaining outliers values in this dataset.

## Part 3: Perform Data Visualization

### Histogram plot Visualization

```
# Histogram for numeric columns
df_clean.hist(figsize=(12, 10), bins=20)
plt.suptitle("Distribution of Numeric Features", fontsize=14)
plt.show()
```

## Distribution of Numeric Features

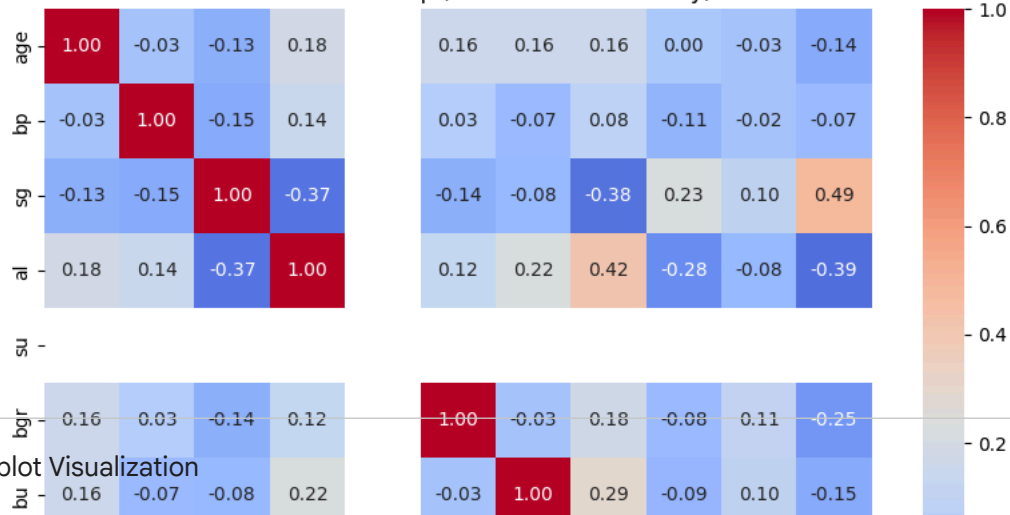


### Correlation heatmap Visualization

```
numeric_df = df_clean.select_dtypes(include=['int64', 'float64'])

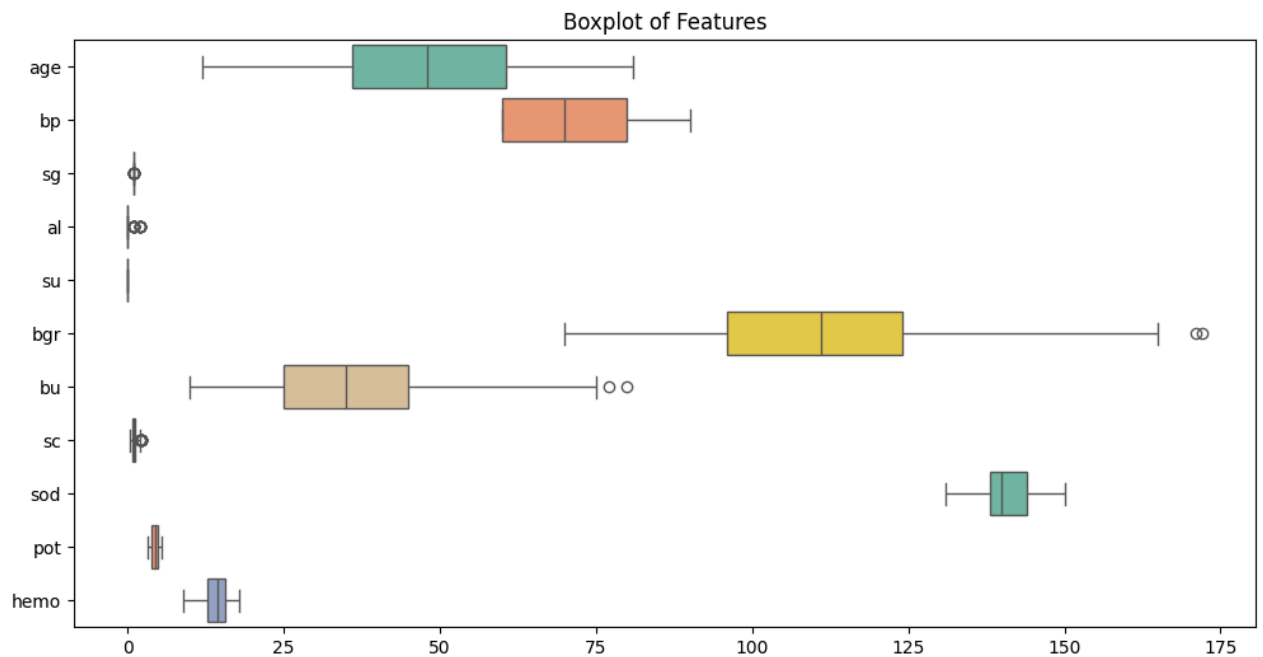
if not numeric_df.empty:
    plt.figure(figsize=(10, 8))
    sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
    plt.title("Correlation Heatmap (Numeric Features Only)")
    plt.show()
else:
    print("No numeric columns available for correlation heatmap.")
```

Correlation Heatmap (Numeric Features Only)



### Boxplot Visualization

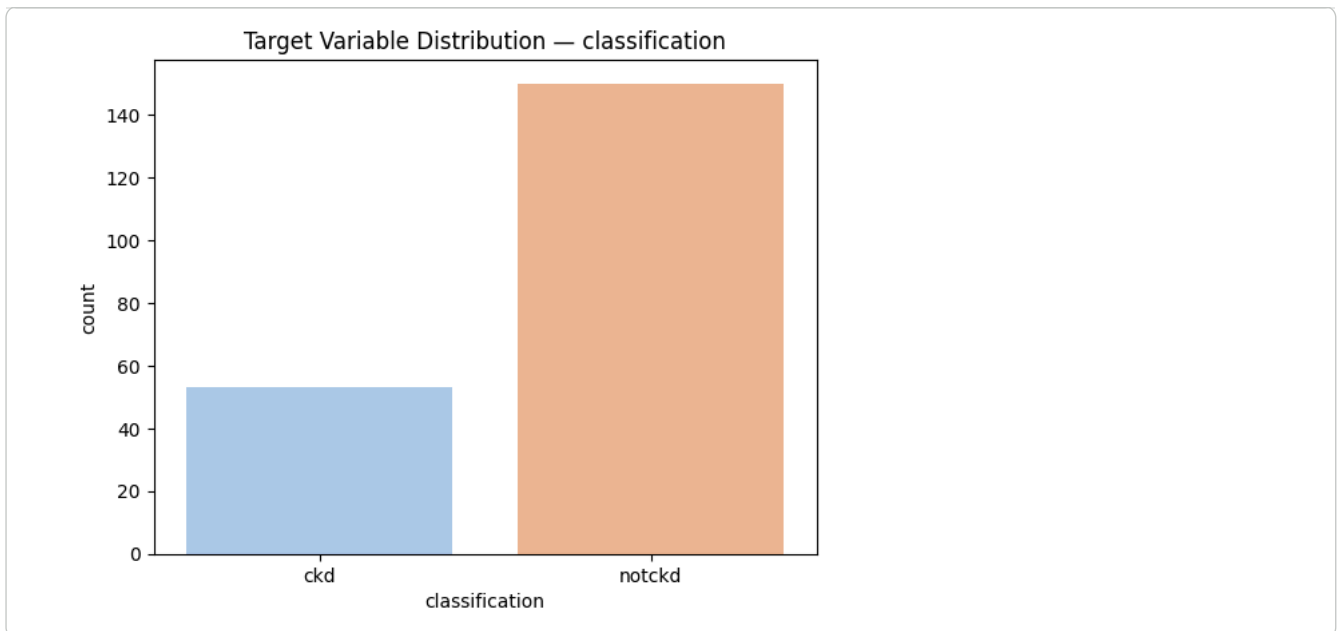
```
# Boxplots to visualize outliers
plt.figure(figsize=(12, 6))
sns.boxplot(data=df_clean, orient="h", palette="Set2")
plt.title("Boxplot of Features")
plt.show()
```



### Target Distribution Plot

```
# --- 4. Target variable distribution (auto-detect) ---
target_candidates = ['target', 'output', 'classification', 'disease', 'label']
target_col = next((col for col in target_candidates if col in df_clean.columns), None)

if target_col:
    sns.countplot(x=target_col, hue=target_col, data=df_clean, palette='pastel', legend=False)
    plt.title(f"Target Variable Distribution - {target_col}")
    plt.show()
else:
    print("⚠️ No target-like column found.")
```



Result: These are different types of visualizations in this dataset.

✅ All preprocessing steps completed successfully — missing values handled, outliers removed, and visualizations generated gracefully.