numpypdpro

July 1, 2024

[72]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[12]:
```
kidney_df=pd.read_csv('kidney-stone-detection.csv')
```

[14]:
```
kidney_df
```

[14]:

	gravity	ph	osmolality	conductivity	urea	calcium	target
0	1.021	4.91	725	14.00	443	2.45	0
1	1.017	5.74	577	20.00	296	4.49	0
2	1.008	7.20	321	14.90	101	2.36	0
3	1.011	5.51	408	12.60	224	2.15	0
4	1.005	6.52	187	7.50	91	1.16	0
..	…	…	…	…	…	…	…
97	1.090	6.19	379	11.28	196	2.50	1
98	1.101	6.92	452	21.49	133	1.27	1
99	1.122	6.11	452	20.70	408	4.18	1
100	1.121	6.02	375	11.01	170	3.10	1
101	1.151	5.61	886	17.52	528	3.01	1

[102 rows x 7 columns]

[17]:
```
kidney_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102 entries, 0 to 101
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   gravity       102 non-null    float64
 1   ph            102 non-null    float64
 2   osmolality    102 non-null    int64  
 3   conductivity  102 non-null    float64
 4   urea          102 non-null    int64  
 5   calcium       102 non-null    float64
 6   target        102 non-null    int64  
```

```
dtypes: float64(4), int64(3)
memory usage: 5.7 KB
```

[22]:
```
kidney_df.describe()
```

[22]:
```
          gravity          ph   osmolality  conductivity        urea  \
count  102.000000  102.000000   102.000000    102.000000  102.000000
mean     1.025830    6.010476   608.823529     20.797371  263.764706
std      0.025374    0.690067   234.320227      7.530517  133.335665
min      1.005000    4.760000   187.000000      5.100000   10.000000
25%      1.013201    5.536520   418.750000     14.650000  159.000000
50%      1.020000    5.905000   577.000000     21.391198  253.500000
75%      1.025000    6.365000   780.500000     26.100000  366.250000
max      1.151000    7.940000  1236.000000     38.000000  620.000000

          calcium      target
count  102.000000  102.000000
mean     4.252362    0.558824
std      3.193619    0.498980
min      0.170000    0.000000
25%      1.482500    0.000000
50%      3.331046    1.000000
75%      6.153294    1.000000
max     14.340000    1.000000
```

[26]:
```
kidney_df.columns
```

[26]:
```
Index(['gravity', 'ph', 'osmolality', 'conductivity', 'urea', 'calcium',
       'target'],
      dtype='object')
```

[28]:
```
kidney_df.shape
```

[28]:
```
(102, 7)
```

[31]:
```
type(kidney_df)
```

[31]:
```
pandas.core.frame.DataFrame
```

[40]:
```
count_ones = kidney_df['target'].value_counts()[1]
print(count_ones)
```

```
57
```

[42]:
```
count_zeroes = kidney_df['target'].value_counts()[0]
print(count_zeroes)
```

```
45
```

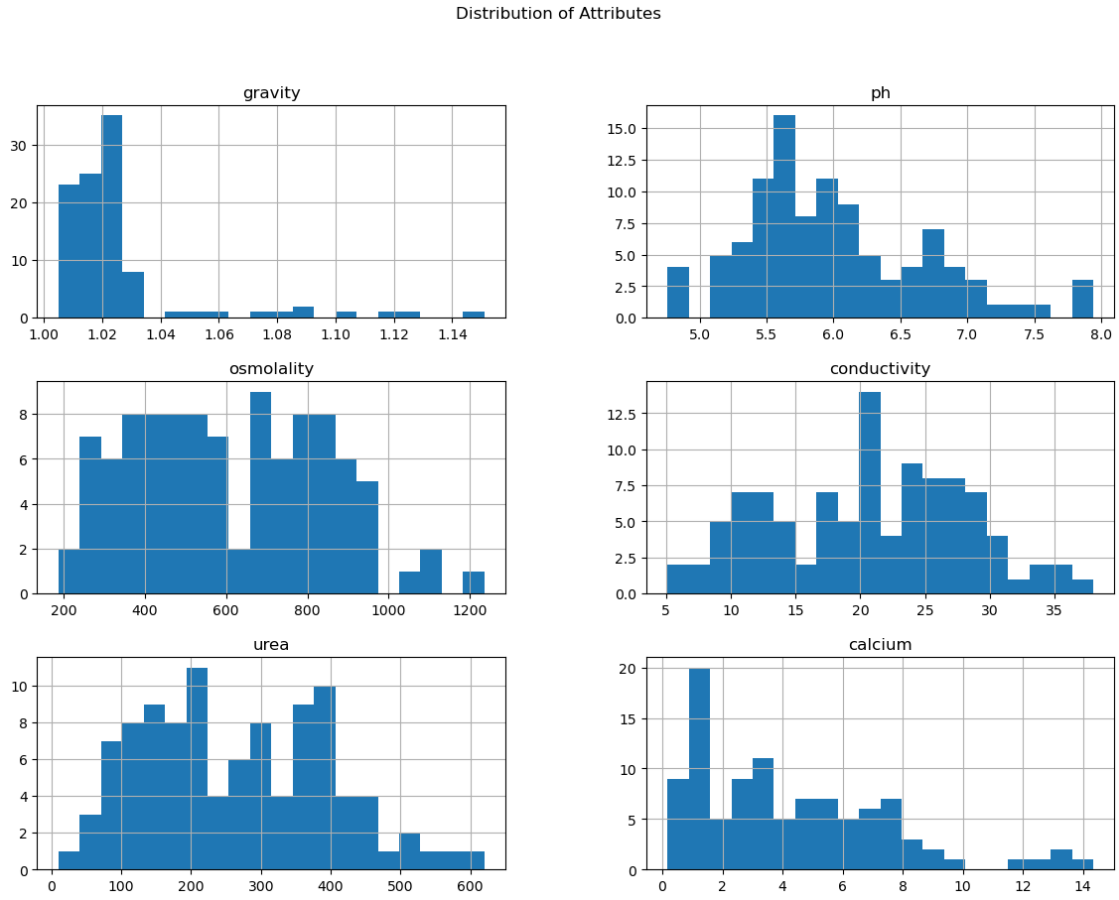
[51]:
```
kidney_df['target'].count()
```

[51]: 102

[59]:
```
stone_detected_rate=((count_ones)/(kidney_df['target'].count()))*100
print('kidney stone detected rate is:',stone_detected_rate)
```

```
kidney stone detected rate is: 55.88235294117647
```

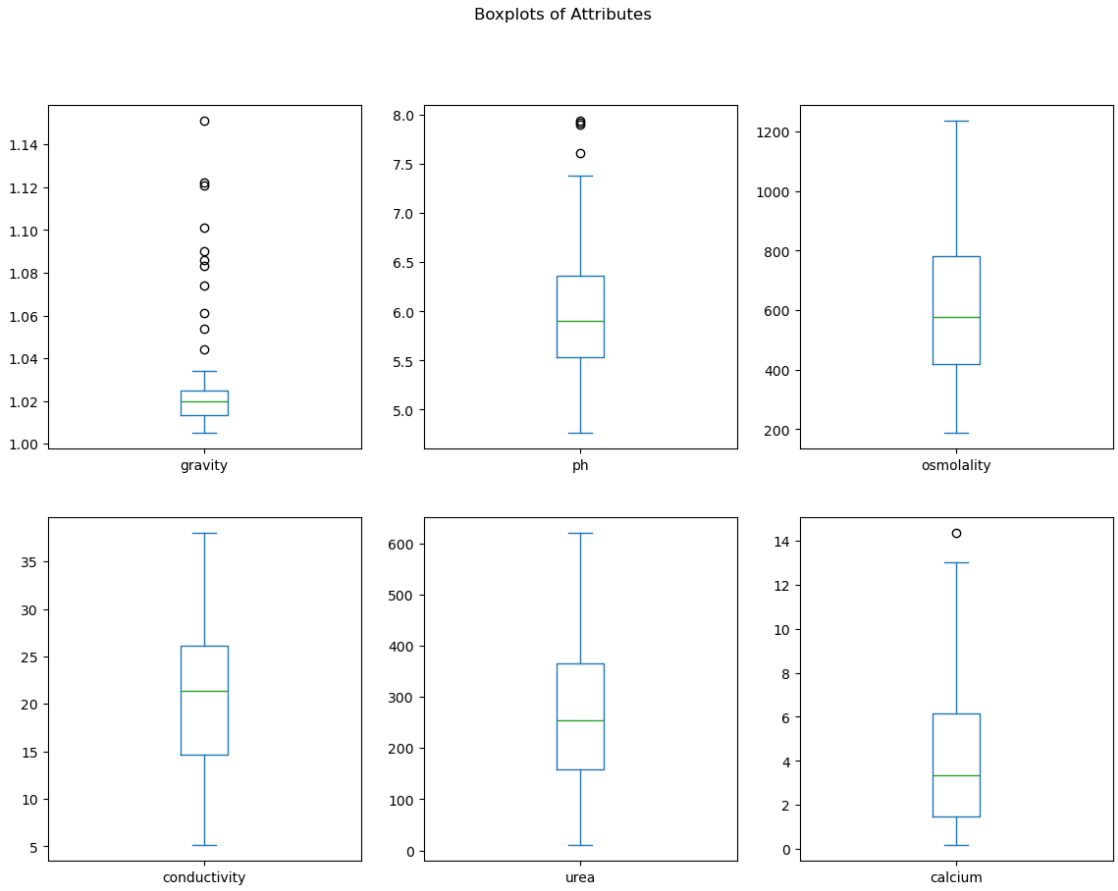
[64]:
```
attributes = ['gravity', 'ph', 'osmolality', 'conductivity', 'urea', 'calcium']

kidney_df[attributes].hist(bins=20, figsize=(14, 10))
plt.suptitle('Distribution of Attributes')
plt.show()
```



Questionnaire for histogram visualization:

1. What is the distribution of each urine test attribute (gravity, pH, osmolality, conductivity, urea, calcium) in the dataset? 2. Are there any attributes with a skewed distribution? 3. Do any attributes have a wide range of values, indicating high variability? 4. Are there any attributes with potential outliers based on their distribution?

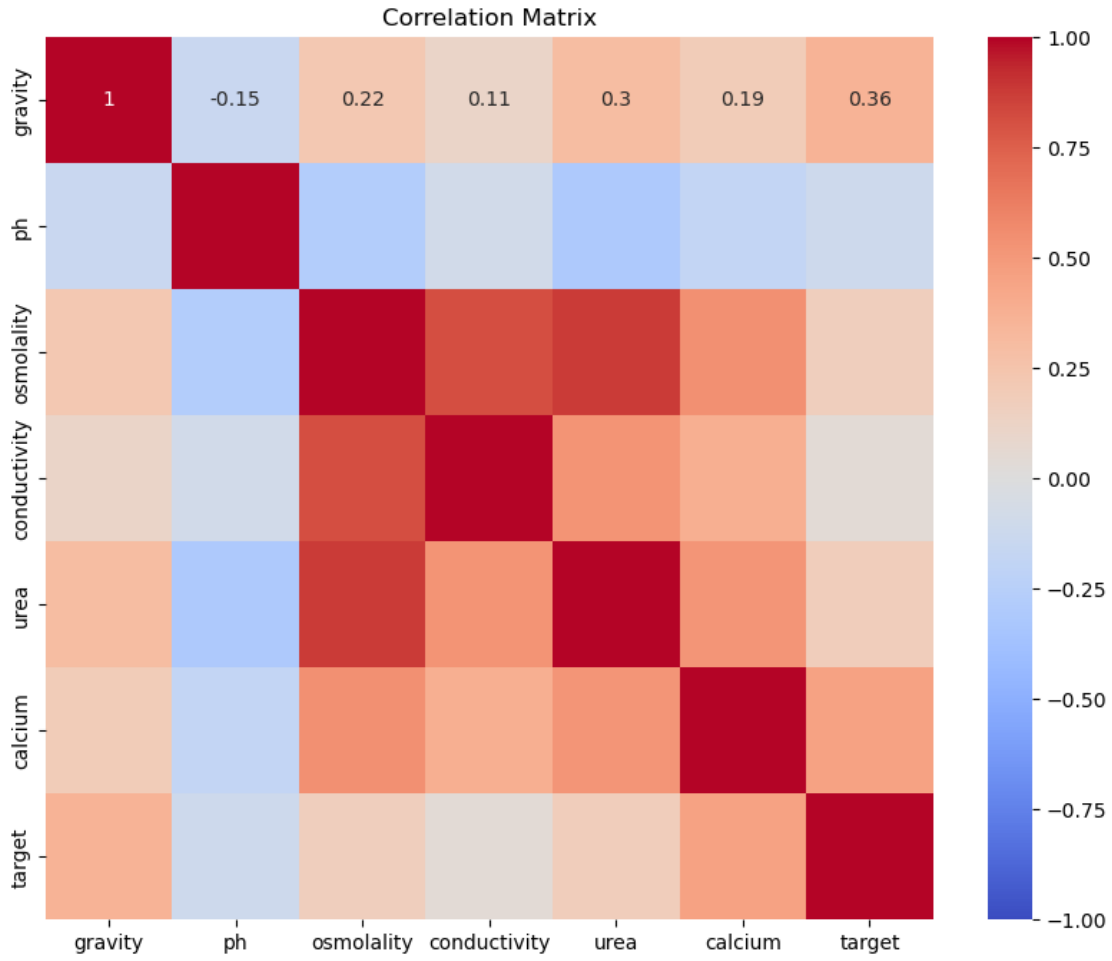
[68]:
```
kidney_df[attributes].plot(kind='box', subplots=True, layout=(2, 3),␣
↪figsize=(14, 10), sharex=False, sharey=False)
plt.suptitle('Boxplots of Attributes')
plt.show()
```



Questionnaire for Boxplot:

1. What is the central tendency (median) of each attribute? 3. How much variability (IQR) is there in the values of each attribute? 4. Are there any outliers in the attributes? 5. How do the values of each attribute spread around the median? 6. Are there any attributes that show a significant difference in distribution when compared side by side?

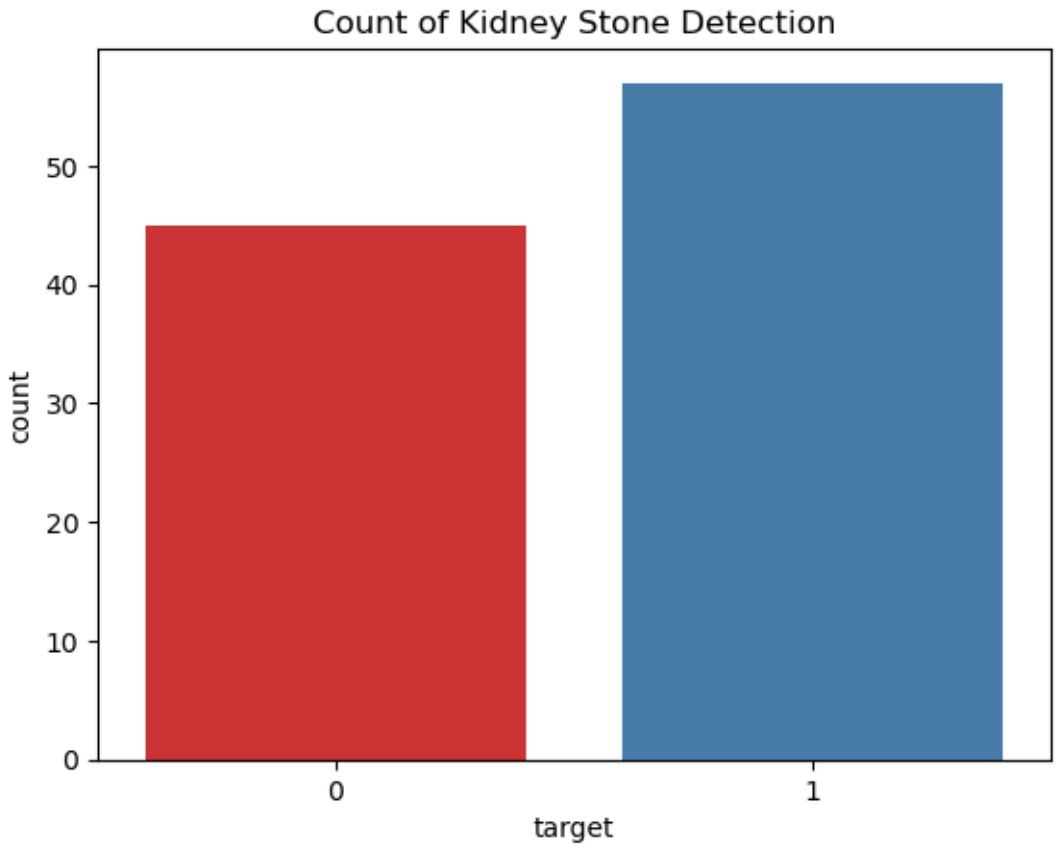
[74]:
```
correlation_matrix = kidney_df.corr()

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Matrix')
plt.show()
```



Questionnaire Correlation Matrix:

1. Which attributes are strongly correlated with each other? 2. Are there any pairs of attributes with a strong positive correlation (close to 1)? 3. Are there any pairs of attributes with a strong negative correlation (close to -1)? 4. Are there attributes that show little to no correlation with others (close to 0)? 5. How might these correlations impact the detection of kidney stones?

[76]:
```
sns.countplot(x='target', data=kidney_df, palette='Set1')
plt.title('Count of Kidney Stone Detection')
plt.show()
```



Questionnaire for Countplot:

1. What is the distribution of the target variable? 2. How many cases in the dataset are identified as having kidney stones (target=1) versus not having kidney stones (target=0)? 3. Is there a significant imbalance between the number of positive and negative cases?

[ ]: