```
7import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
Start coding or generate with AI.
```

```
sns.set(style="whitegrid")
```

```
titanic = sns.load_dataset('diamonds')
titanic.head()
```

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

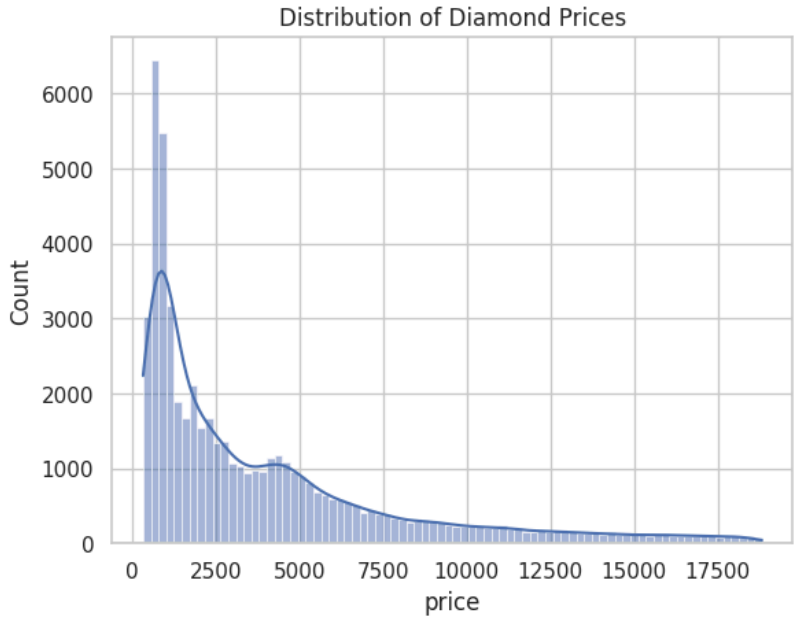
```
df = sns.load_dataset("diamonds")
```

```
df.info()
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 10 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   carat    53940 non-null  float64
 1   cut      53940 non-null  category
 2   color    53940 non-null  category
 3   clarity  53940 non-null  category
 4   depth    53940 non-null  float64
 5   table    53940 non-null  float64
 6   price    53940 non-null  int64
 7   x        53940 non-null  float64
 8   y        53940 non-null  float64
 9   z        53940 non-null  float64
dtypes: category(3), float64(6), int64(1)
memory usage: 3.0 MB
```

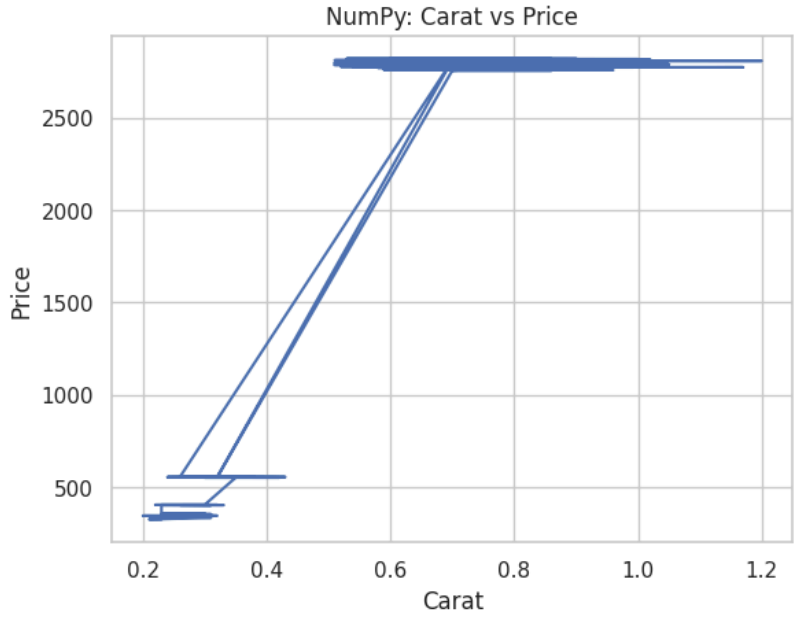
	carat	depth	table	price	x	y	z
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

```
sns.histplot(df['price'], kde=True)
plt.title("Distribution of Diamond Prices")
plt.show()
```

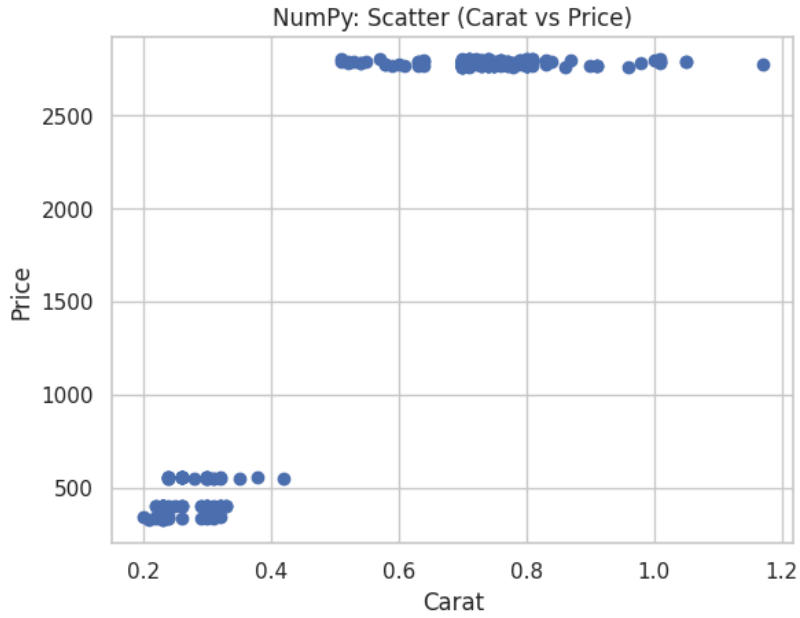



```
x = df['carat'].to_numpy()
y = df['price'].to_numpy()

plt.plot(x[:500], y[:500])
plt.title("NumPy: Carat vs Price")
plt.xlabel("Carat")
plt.ylabel("Price")
plt.show()
```

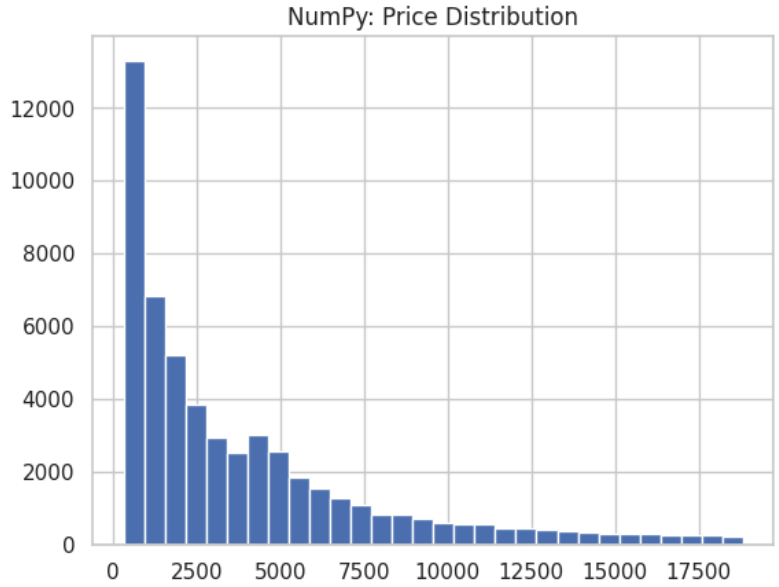



```
plt.scatter(x[:300], y[:300])
plt.title("NumPy: Scatter (Carat vs Price)")
plt.xlabel("Carat")
plt.ylabel("Price")
plt.show()
```



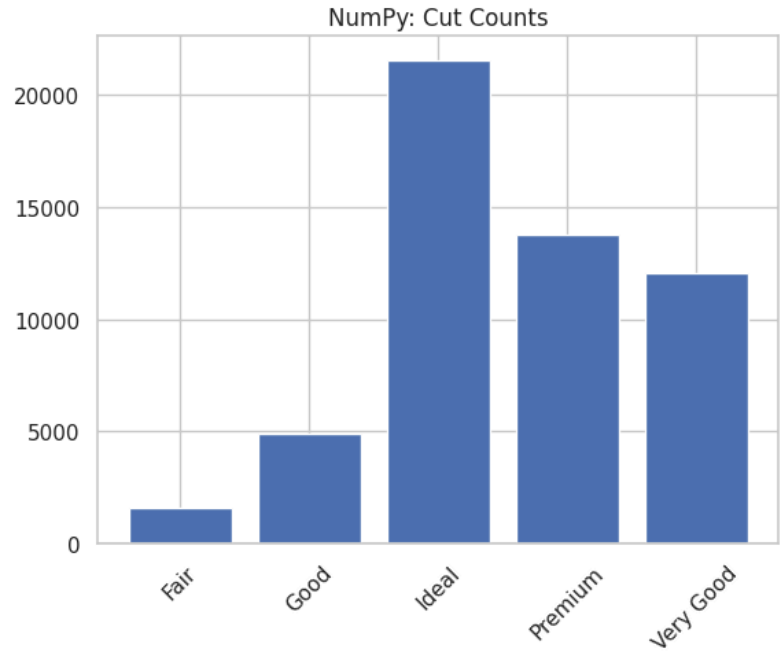

```
prices = df['price'].to_numpy()

plt.hist(prices, bins=30)
plt.title("NumPy: Price Distribution")
plt.show()
```



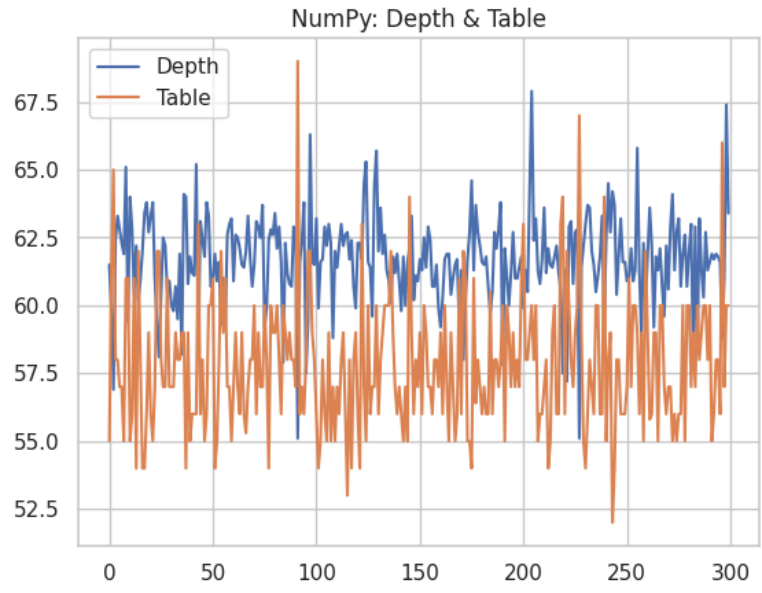

```
cuts, counts = np.unique(df['cut'].to_numpy(), return_counts=True)

plt.bar(cuts, counts)
plt.title("NumPy: Cut Counts")
plt.xticks(rotation=45)
plt.show()
```

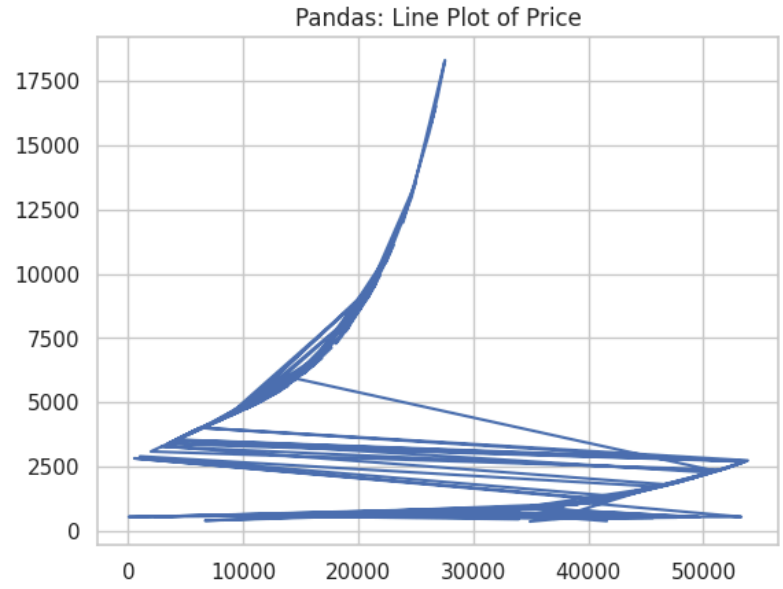



```
depth = df['depth'].to_numpy()[:300]
table = df['table'].to_numpy()[:300]

plt.plot(depth, label="Depth")
plt.plot(table, label="Table")
plt.title("NumPy: Depth & Table")
plt.legend()
plt.show()
```

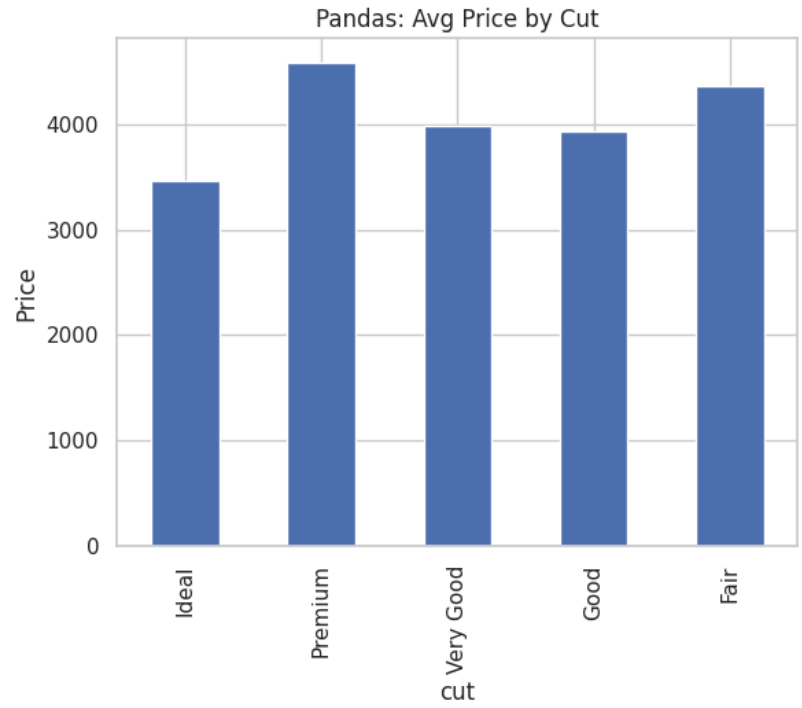



```
df.sample(200).sort_values("carat")['price'].plot(kind='line')
plt.title("Pandas: Line Plot of Price")
plt.show()
```

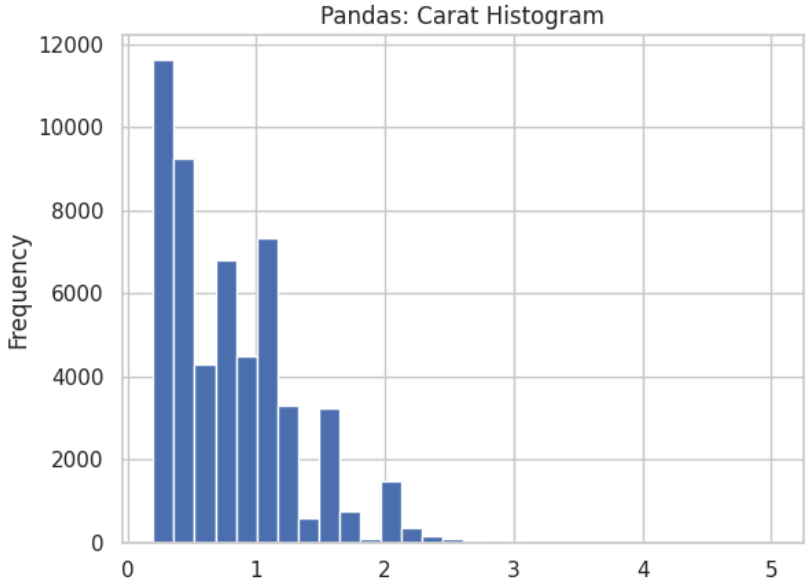



```
df.groupby('cut')['price'].mean().plot(kind='bar')
plt.title("Pandas: Avg Price by Cut")
plt.ylabel("Price")
plt.show()
```

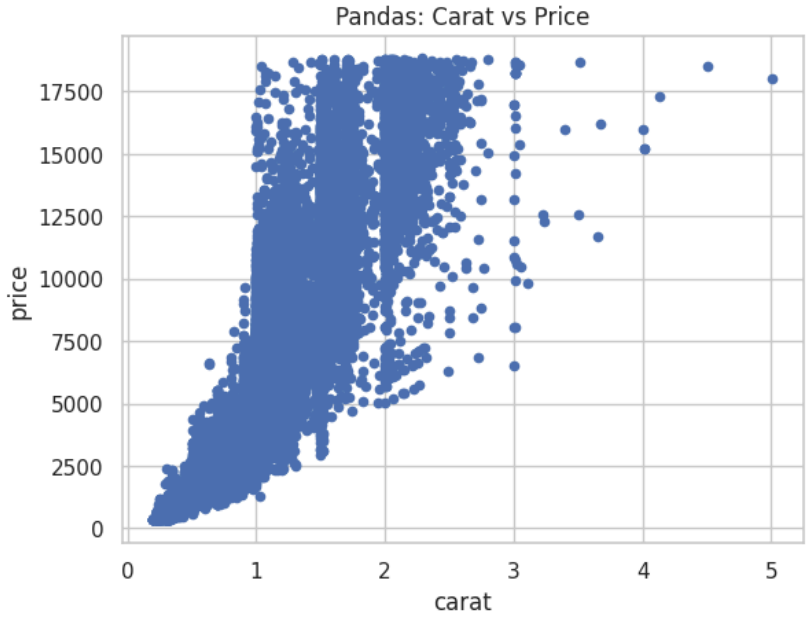
```
/tmp/ipython-input-1190515874.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to Tr
  df.groupby('cut')['price'].mean().plot(kind='bar')
```



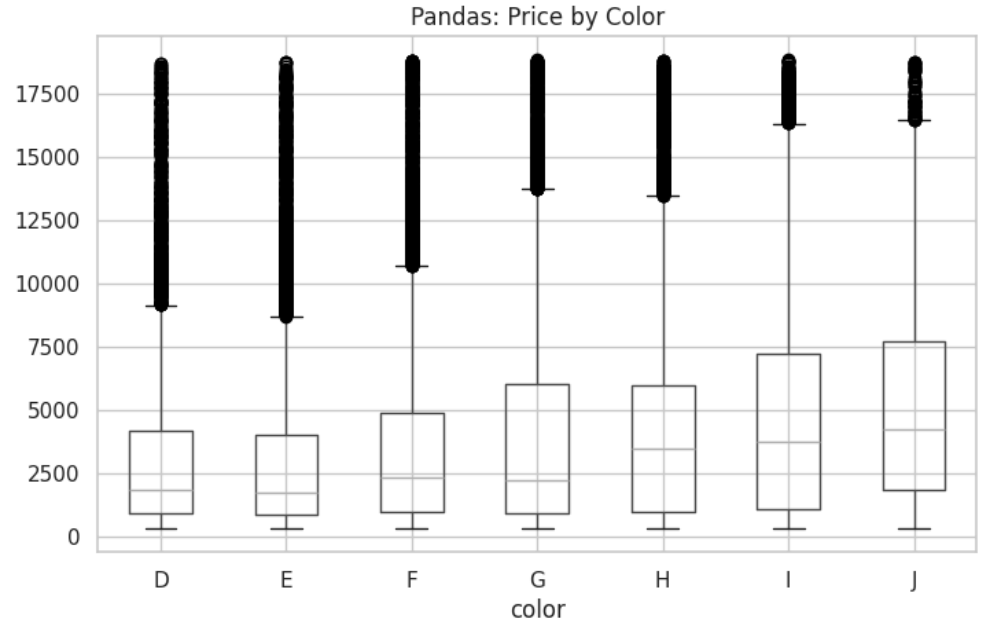

```
df['carat'].plot(kind='hist', bins=30)
plt.title("Pandas: Carat Histogram")
plt.show()
```



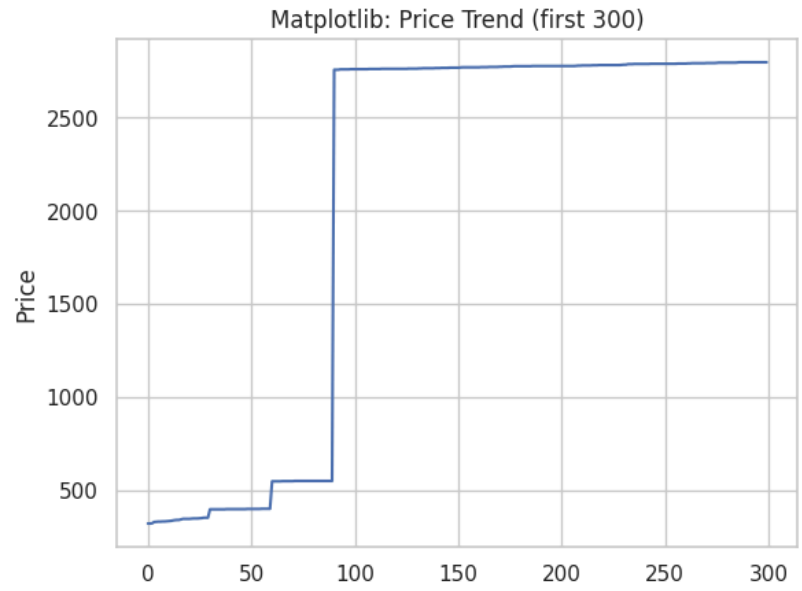

```
df.plot(kind='scatter', x='carat', y='price')
plt.title("Pandas: Carat vs Price")
plt.show()
```




```
df.boxplot(column='price', by='color', figsize=(8,5))
plt.title("Pandas: Price by Color")
plt.suptitle("")
plt.show()
```

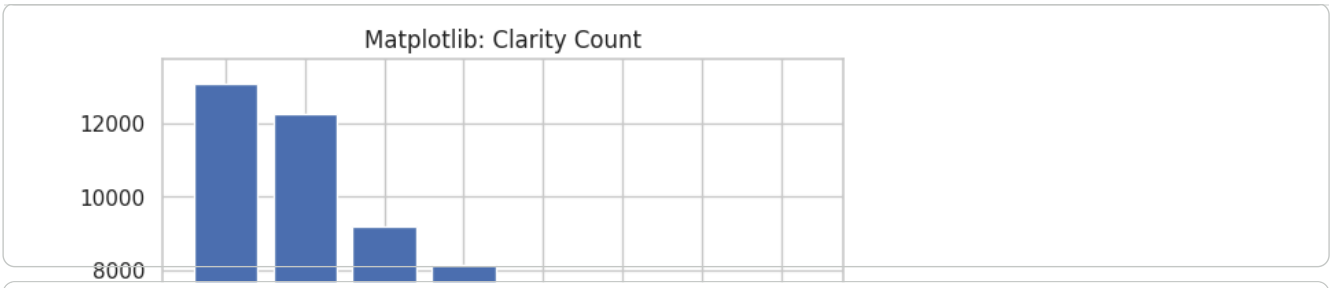



```
plt.plot(df['price'][:300])
plt.title("Matplotlib: Price Trend (first 300)")
plt.ylabel("Price")
plt.show()
```

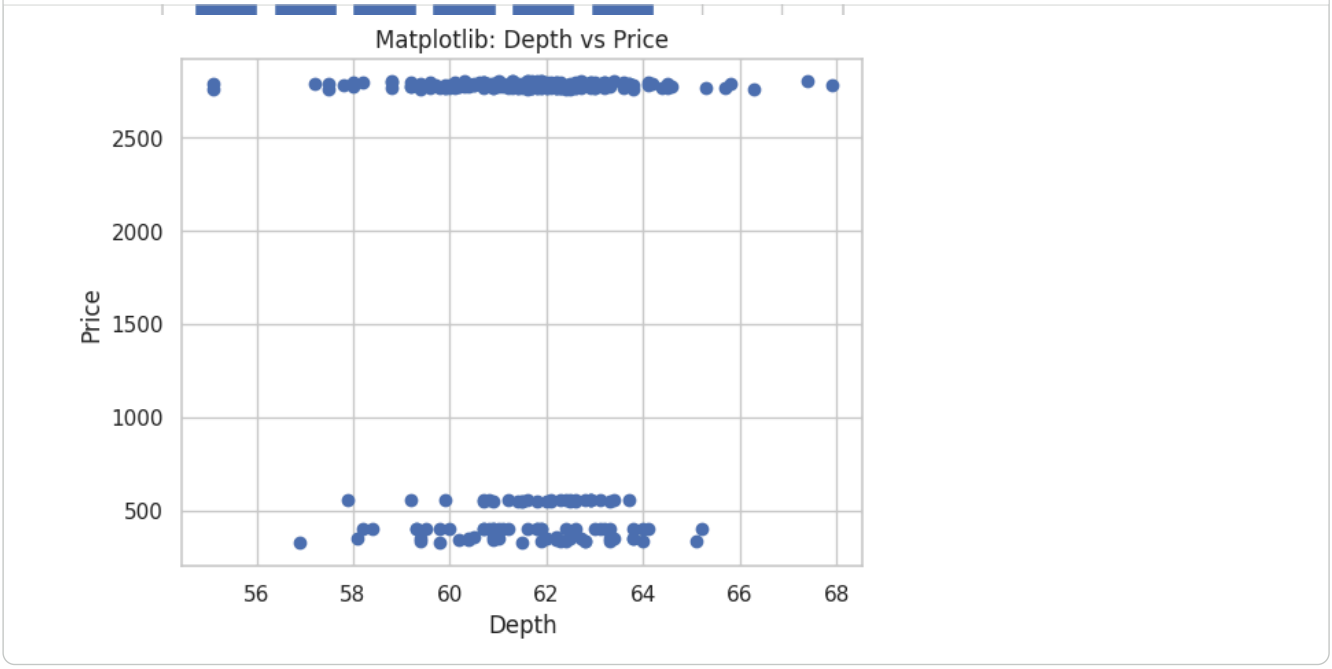



```
clarity_counts = df['clarity'].value_counts()

plt.bar(clarity_counts.index, clarity_counts.values)
plt.title("Matplotlib: Clarity Count")
plt.xticks(rotation=45)
plt.show()
```




```
plt.scatter(df['depth'][:300], df['price'][:300])
plt.title("Matplotlib: Depth vs Price")
plt.xlabel("Depth")
plt.ylabel("Price")
plt.show()
```




```
cut_counts = df['cut'].value_counts()

plt.pie(cut_counts.values, labels=cut_counts.index, autopct="%1.1f%%")
plt.title("Matplotlib: Cut Distribution")
plt.show()
```