Seaborn

- is also a data visualization library which is build on top of matplotlib.

In [18]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn  as sns     # import seaborn library
```

Load the dataset from seaborns *load_dataset* module

In [10]:
```
sns.get_dataset_names()    # it will give you the datasets name
```

Out[10]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
 'fmri',
 'gammas',
 'geyser',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'taxis',
 'tips',
 'titanic']
```

In [17]:
```
df = sns.load_dataset('diamonds')
df
```

Out[17]:

	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
...	...	...	...	...	...	...	...	...	...	...
53935	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 10 columns

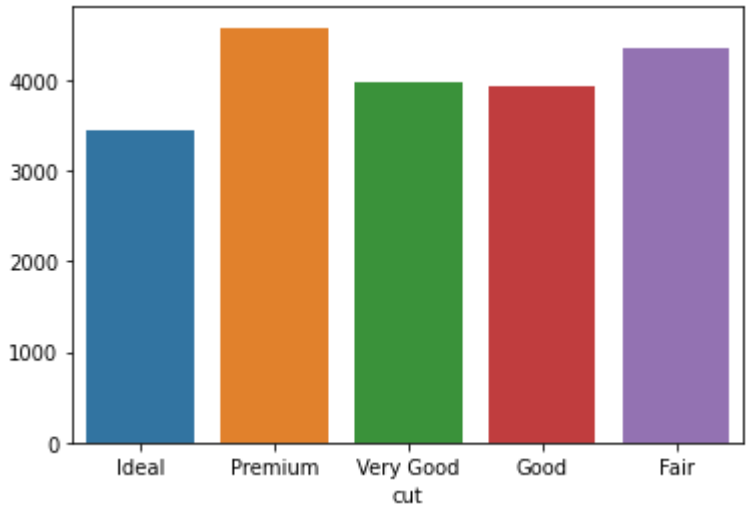
Bar Plots

In [19]:
```
df.cut.value_counts()
```

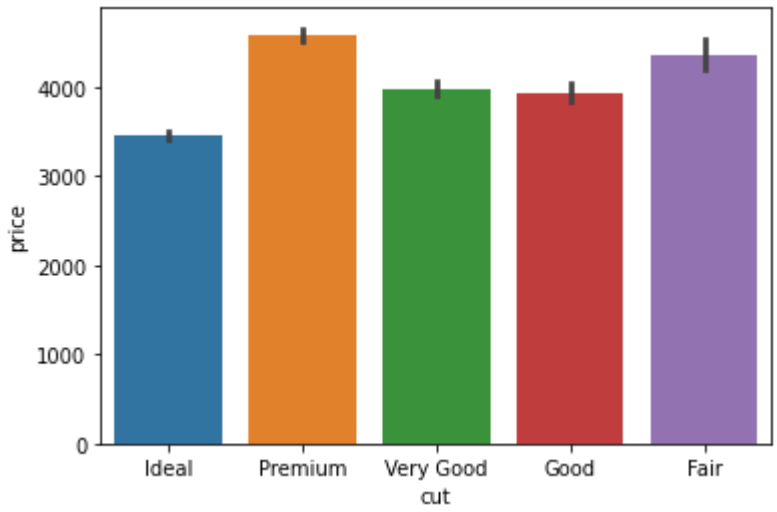
Out[19]:
```
Ideal        21551
Premium      13791
Very Good    12082
Good          4906
Fair          1610
Name: cut, dtype: int64
```

In [23]:
```
cut_price = df.groupby(by = 'cut')['price'].mean()

x = cut_price.index
y = cut_price.values
```

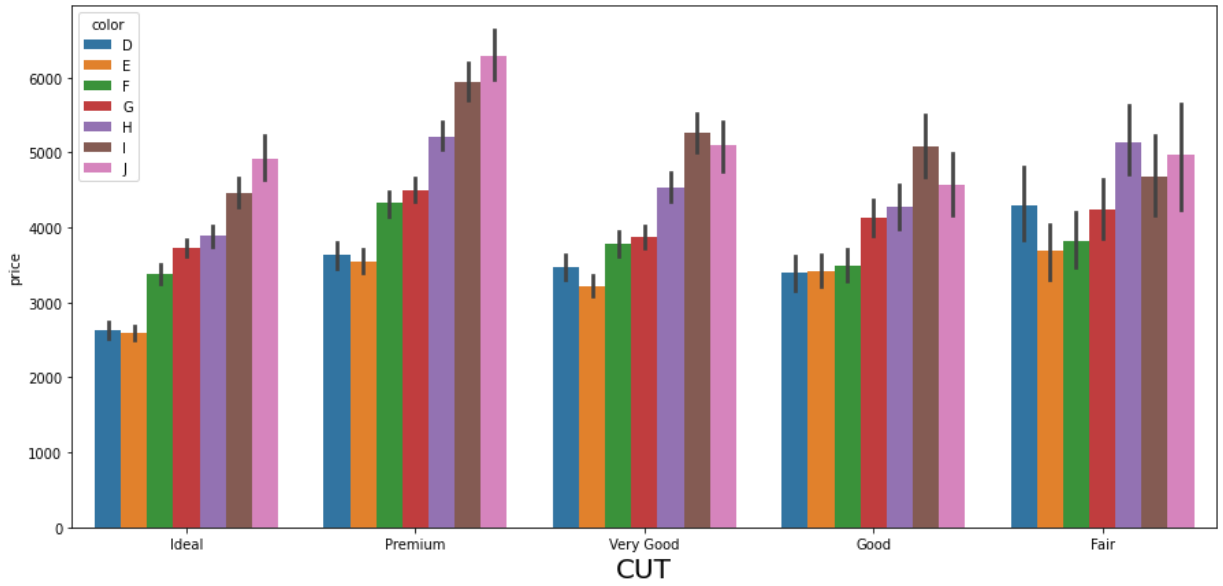
In [25]:
```
1 sns.barplot(x=x, y = y)
2 plt.show()
```



In [28]:
```
1 # second method
2
3 sns.barplot(x = 'cut', y = 'price',data = df)
4 plt.show()
```



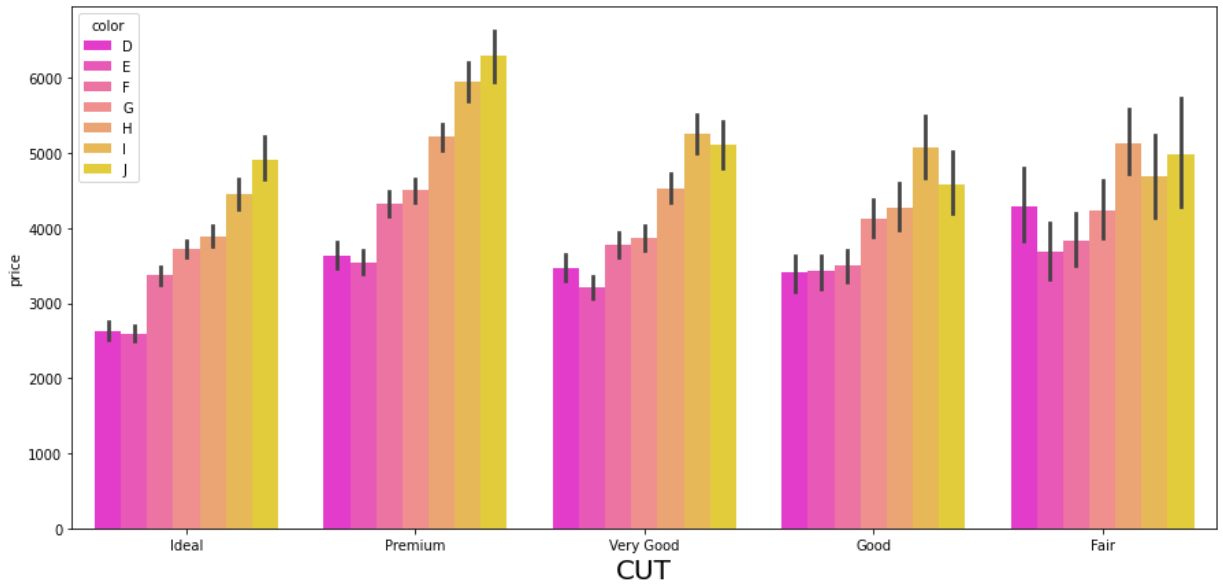
In [50]:
```
plt.figure(figsize = (15,7))
sns.barplot(x = 'cut', y = 'price',hue='color',data = df)
plt.xlabel("CUT",size = 20)
plt.show()
```



Color Palettes in Seaborn:

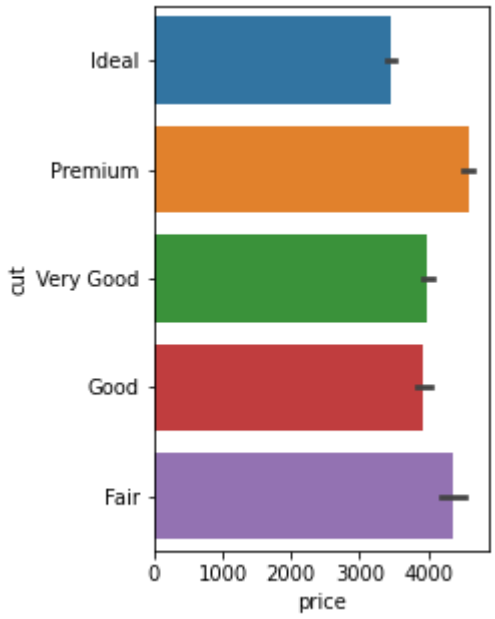
'Accent', 'Accent_r', 'Blues', 'Blues_r', 'BrBG', 'BrBG_r', 'BuGn', 'BuGn_r', 'BuPu', 'BuPu_r', 'CMRmap', 'CMRmap_r', 'Dark2', 'Dark2_r', 'GnBu', 'GnBu_r', 'Greens', 'Greens_r', 'Greys', 'Greys_r', 'OrRd', 'OrRd_r', 'Oranges', 'Oranges_r', 'PRGn', 'PRGn_r', 'Paired', 'Paired_r', 'Pastel1', 'Pastel1_r', 'Pastel2', 'Pastel2_r', 'PiYG', 'PiYG_r', 'PuBu', 'PuBuGn', 'PuBuGn_r', 'PuBu_r', 'PuOr', 'PuOr_r', 'PuRd', 'PuRd_r', 'Purples', 'Purples_r', 'RdBu', 'RdBu_r', 'RdGy', 'RdGy_r', 'RdPu', 'RdPu_r', 'RdYlBu', 'RdYlBu_r', 'RdYlGn', 'RdYlGn_r', 'Reds', 'Reds_r', 'Set1', 'Set1_r', 'Set2', 'Set2_r', 'Set3', 'Set3_r', 'Spectral', 'Spectral_r', 'Wistia', 'Wistia_r', 'YlGn', 'YlGnBu', 'YlGnBu_r', 'YlGn_r', 'YlOrBr', 'YlOrBr_r', 'YlOrRd', 'YlOrRd_r', 'afmhot', 'afmhot_r', 'autumn', 'autumn_r', 'binary', 'binary_r', 'bone', 'bone_r', 'brg', 'brg_r', 'bwr', 'bwr_r', 'cividis', 'cividis_r', 'cool', 'cool_r', 'coolwarm', 'coolwarm_r', 'copper', 'copper_r', 'crest', 'crest_r', 'cubehelix', 'cubehelix_r', 'flag', 'flag_r', 'flare', 'flare_r', 'gist_earth', 'gist_earth_r', 'gist_gray', 'gist_gray_r', 'gist_heat', 'gist_heat_r', 'gist_ncar', 'gist_ncar_r', 'gist_rainbow', 'gist_rainbow_r', 'gist_stern', 'gist_stern_r', 'gist_yarg', 'gist_yarg_r', 'gnuplot', 'gnuplot2', 'gnuplot2_r', 'gnuplot_r', 'gray', 'gray_r', 'hot', 'hot_r', 'hsv', 'hsv_r', 'icefire', 'icefire_r', 'inferno', 'inferno_r', 'jet', 'jet_r', 'magma', 'magma_r', 'mako', 'mako_r', 'nipy_spectral', 'nipy_spectral_r', 'ocean', 'ocean_r', 'pink', 'pink_r', 'plasma', 'plasma_r', 'prism', 'prism_r', 'rainbow', 'rainbow_r', 'rocket', 'rocket_r', 'seismic', 'seismic_r', 'spring', 'spring_r', 'summer', 'summer_r', 'tab10', 'tab10_r', 'tab20', 'tab20_r', 'tab20b', 'tab20b_r', 'tab20c', 'tab20c_r', 'terrain', 'terrain_r', 'turbo', 'turbo_r', 'twilight', 'twilight_r', 'twilight_shifted', 'twilight_shifted_r', 'viridis', 'viridis_r', 'vlag', 'vlag_r', 'winter', 'winter_r'

In [62]:
```
plt.figure(figsize = (15,7))
sns.barplot(x = 'cut', y = 'price',hue='color',data = df, palette = 'spring'
plt.xlabel("CUT",size = 20)
plt.show()
```



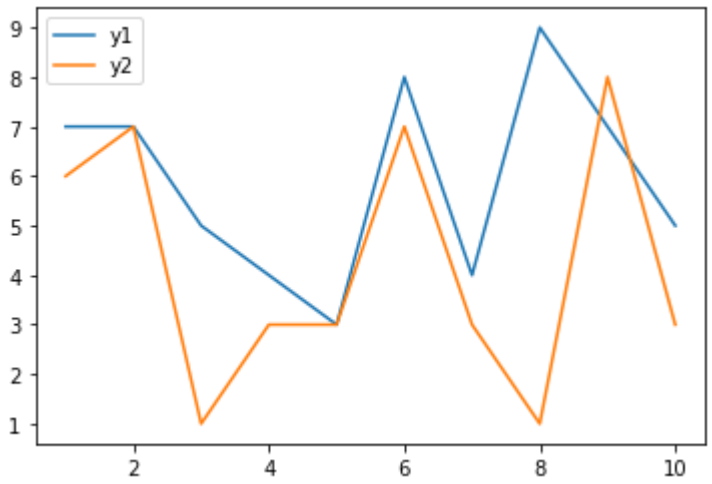
Horizontal Bar Plots

In [48]:
```
plt.figure(figsize = (3,5))
sns.barplot(x = 'price', y = 'cut',data = df, orient = 'h')    # orient = 'v
plt.show()
```



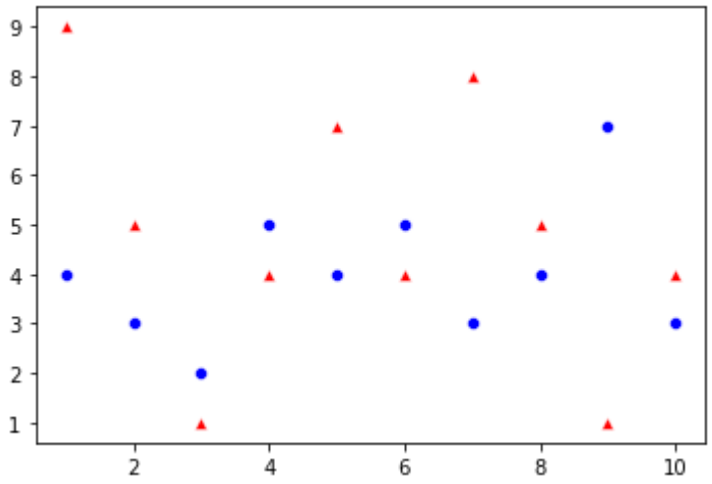
Line Plots

In [70]:
```
x = range(1,11)
y1 = np.random.randint(1,10,10)
y2 = np.random.randint(1,10,10)

sns.lineplot(x = x,y = y1, label = 'y1')
sns.lineplot(x = x,y = y2, label = 'y2')
plt.legend()
plt.show()
```



Scatter Plot

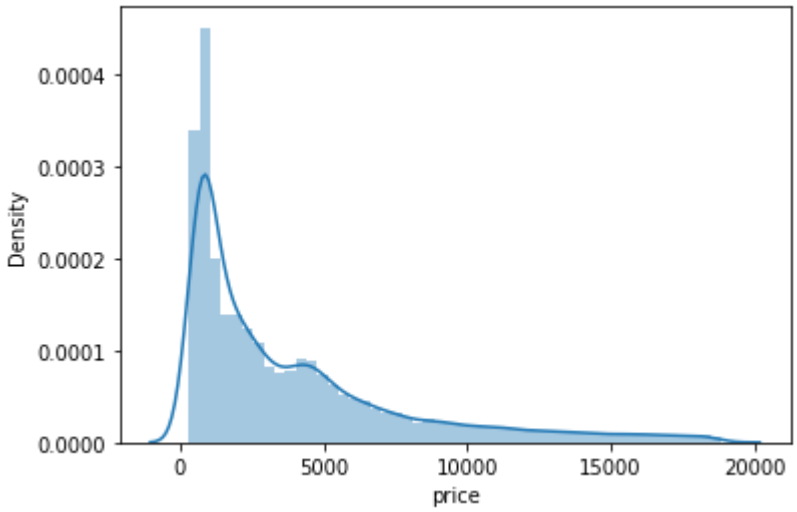
In [98]:
```
x = range(1,11)
y1 = np.random.randint(1,10,10)
y2 = np.random.randint(1,10,10)

sns.scatterplot(x=x,y=y1, color = 'blue')
sns.scatterplot(x=x,y=y2, marker = "^", color = 'red')
plt.show()
```



Distribution Plot

In [104]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [105]:
```
sns.distplot(a = df.price)
plt.show()
```

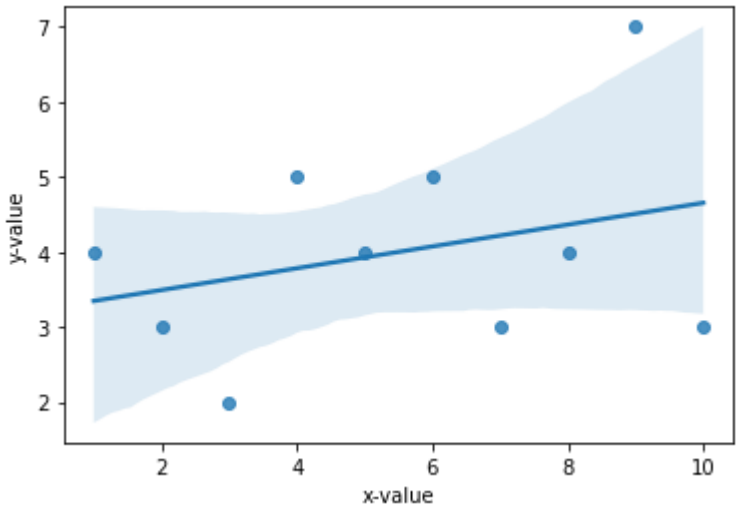


Reg Plot

- fits a best fit line on the dataset

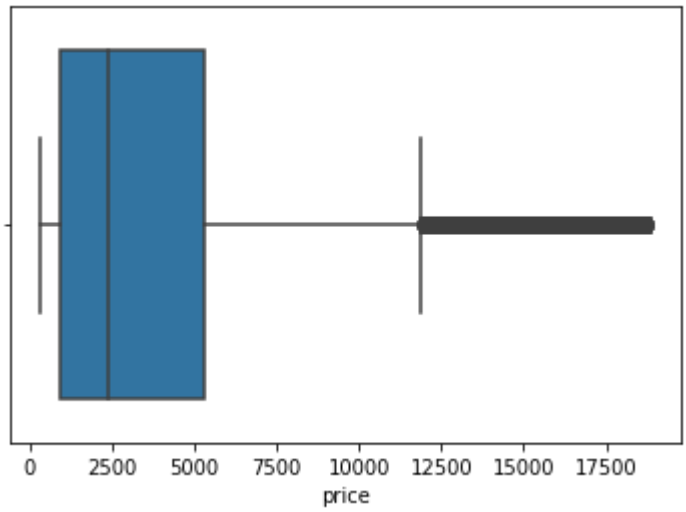
In [112]:
```
x = list(range(1,11))
y = [4, 3, 2, 5, 4, 5, 3, 4, 7, 3]
```

In [114]:
```
sns.regplot(x = x,y = y)
plt.xlabel("x-value")
plt.ylabel("y-value")
plt.show()
```

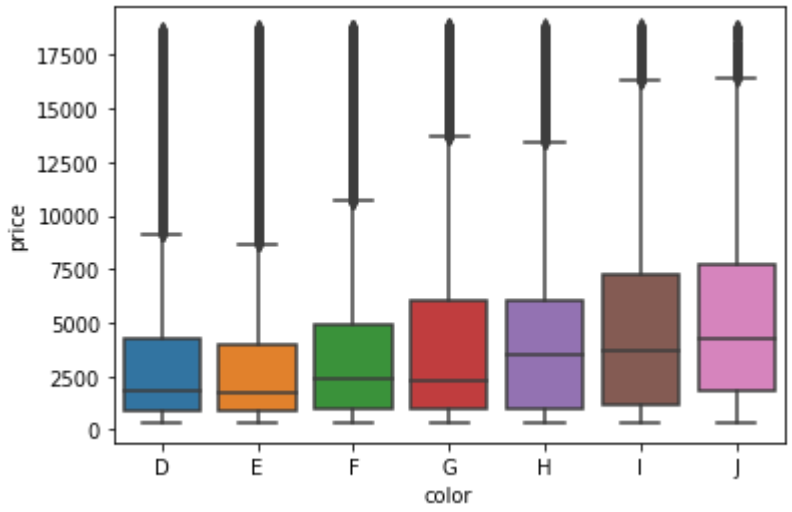


Box Plot

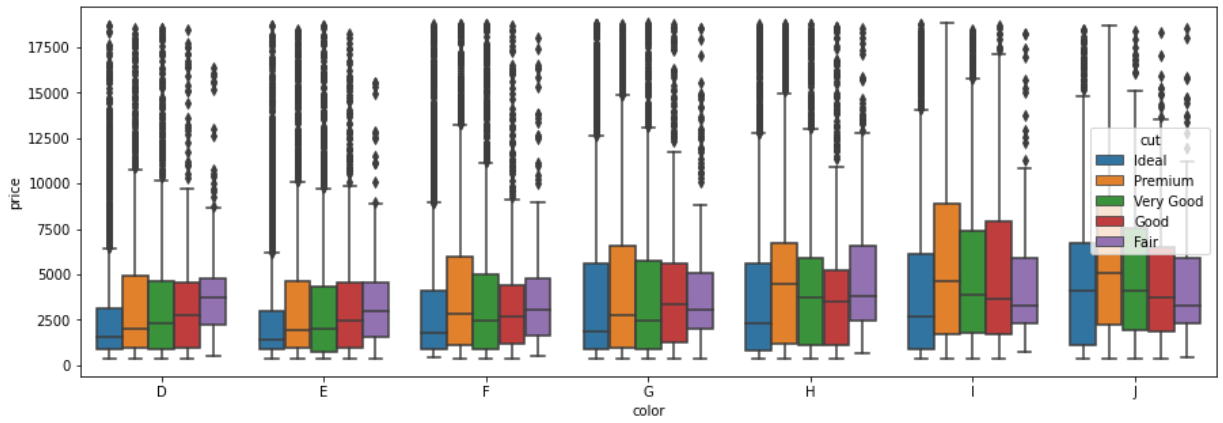
In [116]:
```
1 sns.boxplot(df.price)
2 plt.show()
```



In [124]:
```
1 sns.boxplot(x = 'color', y = 'price',data = df)
2 plt.show()
```



In [127]:
```
plt.figure(figsize = (15,5))
sns.boxplot(x = 'color', y = 'price',hue ="cut",data = df)
plt.show()
```



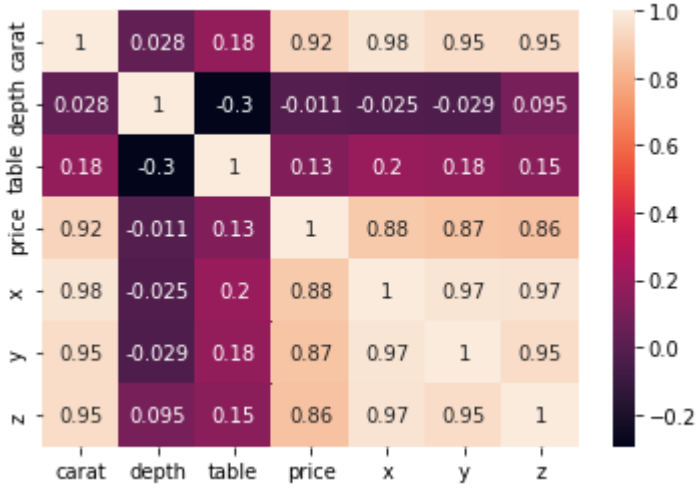
Heatmap

In [128]:
```
df.corr()
```

Out[128]:

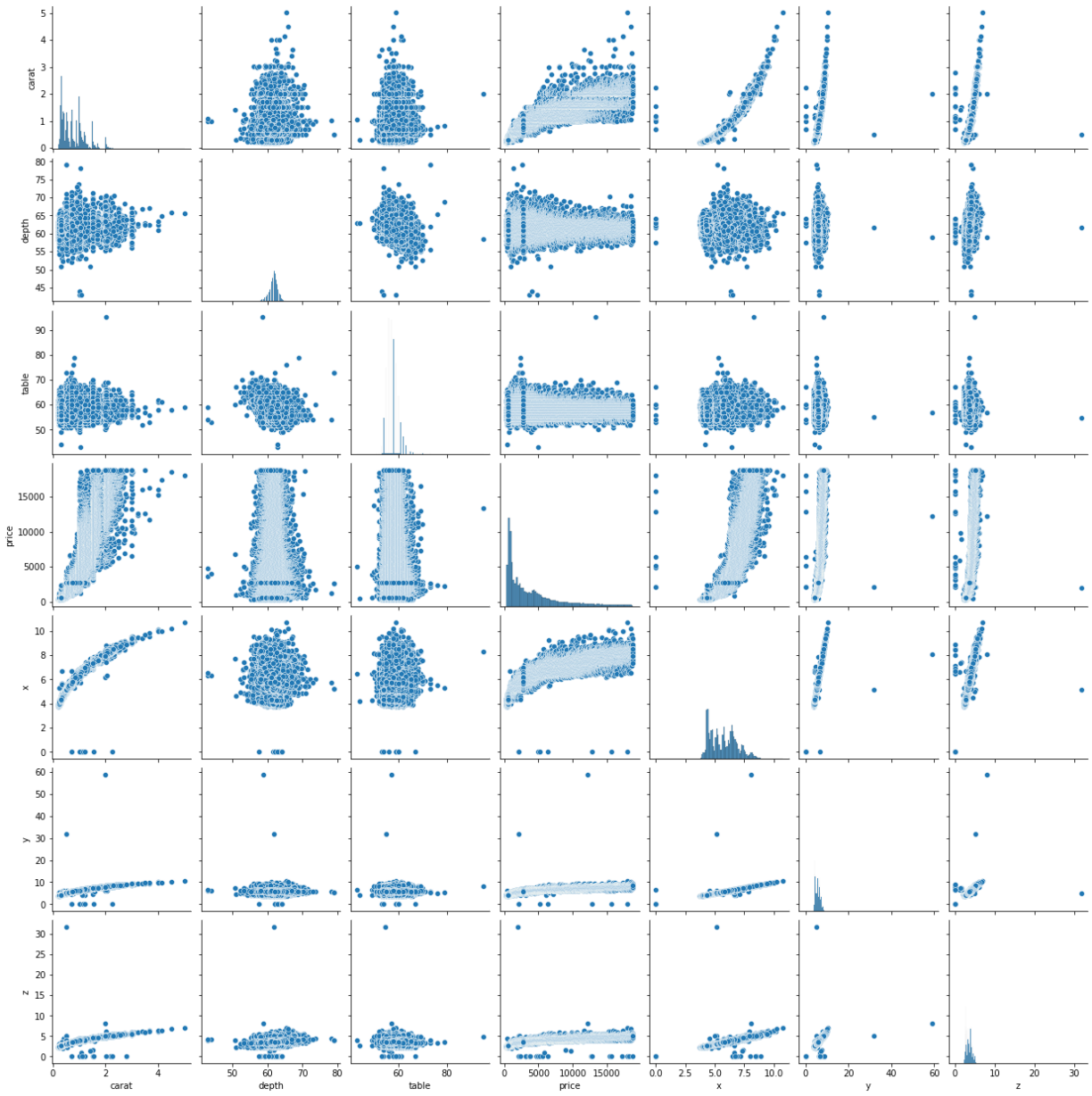
	carat	depth	table	price	x	y	z
carat	1.000000	0.028224	0.181618	0.921591	0.975094	0.951722	0.953387
depth	0.028224	1.000000	-0.295779	-0.010647	-0.025289	-0.029341	0.094924
table	0.181618	-0.295779	1.000000	0.127134	0.195344	0.183760	0.150929
price	0.921591	-0.010647	0.127134	1.000000	0.884435	0.865421	0.861249
x	0.975094	-0.025289	0.195344	0.884435	1.000000	0.974701	0.970772
y	0.951722	-0.029341	0.183760	0.865421	0.974701	1.000000	0.952006
z	0.953387	0.094924	0.150929	0.861249	0.970772	0.952006	1.000000

In [131]:
```
sns.heatmap(df.corr(), annot  = True)
plt.show()
```



Pairplot

In [134]:
```
1 sns.pairplot(df)
2 plt.show()
```



In [ ]:
```
1
```