Análisis de Datos y Aprendizaje Máquina con Tensorflow 2.0: Pre-procesamiento de Datos para Aprendizaje Máquina

2019/09/30

Visualización de datos

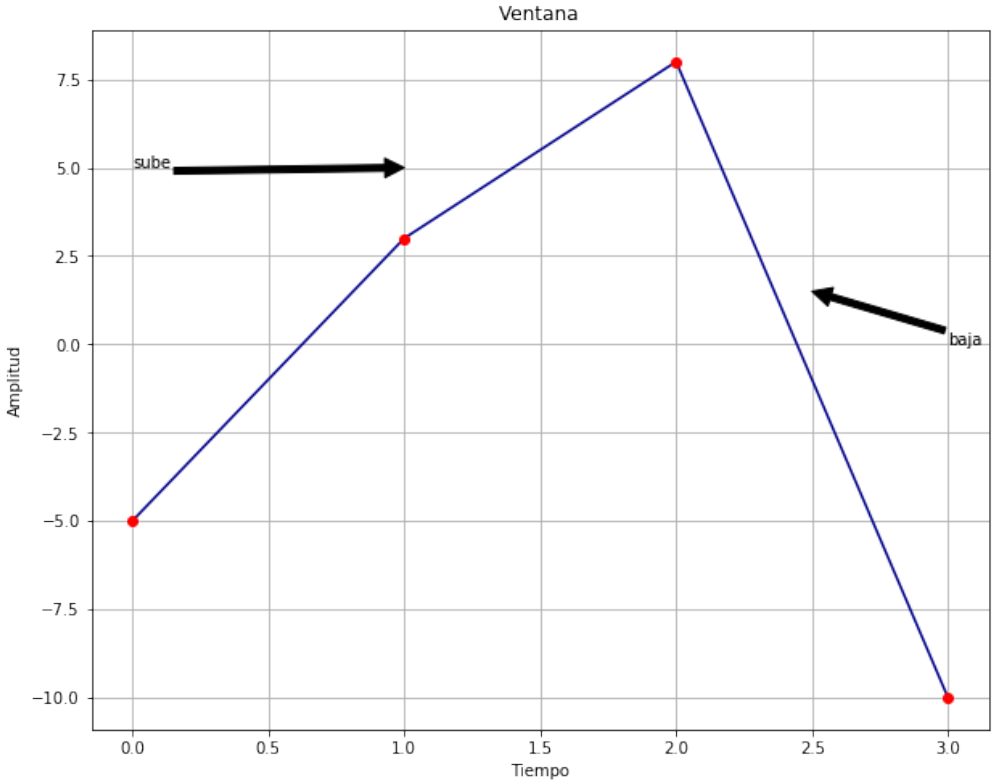
Objetivo: Conocer las herramientas para visualización de datos.

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

Plotear vector

```
In [2]: v3 = np.array([-5, 3, 8, -10])
```

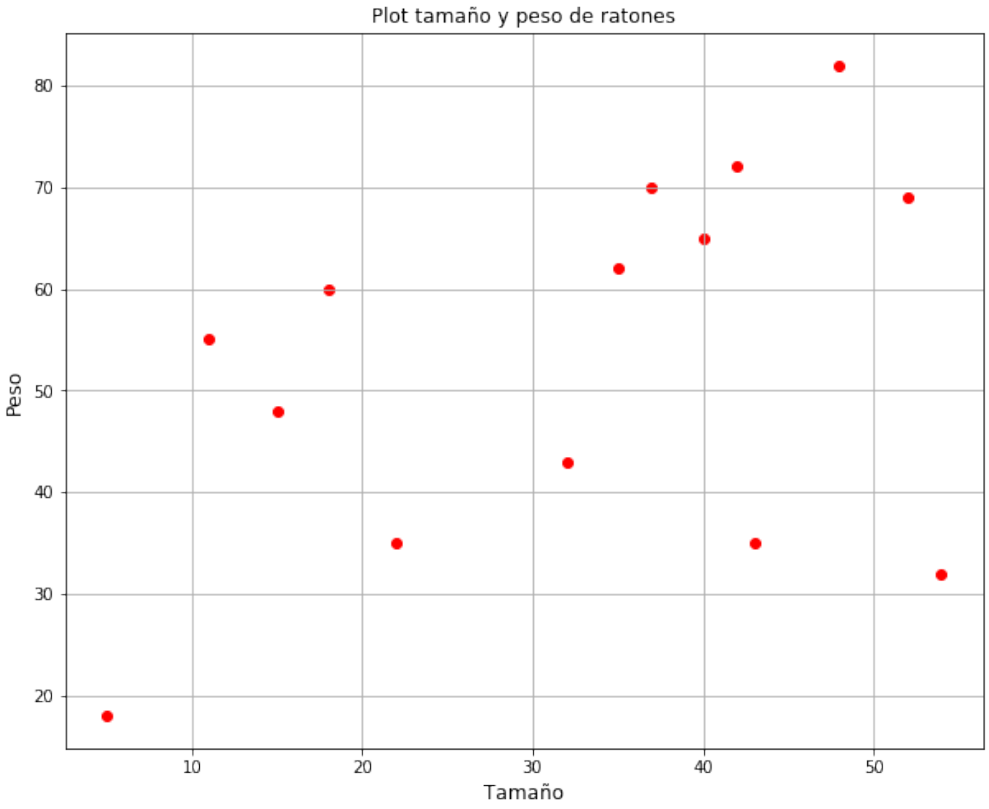
```
In [3]: fig = plt.figure(figsize=(10,8))
        ax = fig.add_subplot(1, 1, 1)
        ax.grid()

        ax.set_xlabel('Tiempo')
        ax.set_ylabel('Amplitud')
        ax.set_title('Ventana')
        ax.annotate('baja', xy=(2.5, 1.5), xytext=(3, 0),
                    arrowprops=dict(facecolor='black'),
                    )
        ax.annotate('sube', xy=(1, 5), xytext=(0, 5),
                    arrowprops=dict(facecolor='black'),
                    )
        plt.plot(v3, "-",c='darkblue')
        plt.plot(v3, "ro")
        plt.show()
```



Plotear variables

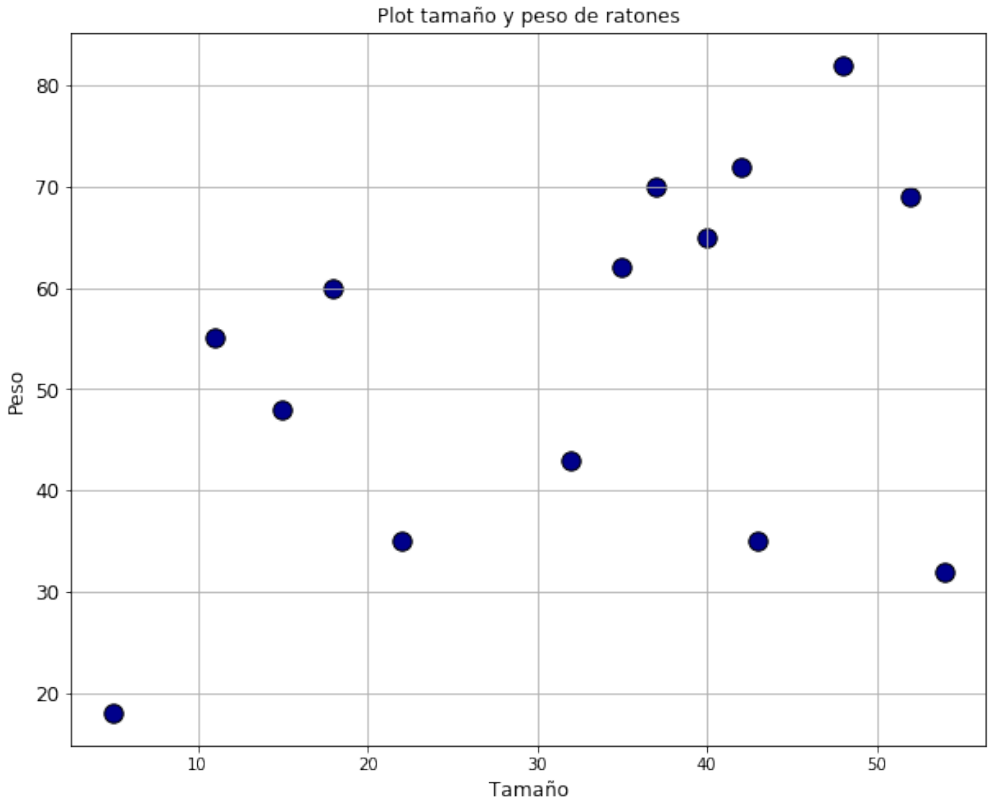
- Plotear pesos de ratones

```
In [4]: tamaño = [11,22,32,35,37,18,42,52,5,40,48,15,43,54]
        peso = [55,35,43,62,70,60,72,69,18,65,82,48,35,32]
        fonts = 12
        plt.figure(figsize=(10,8))
        plt.title("Plot tamaño y peso de ratones",fontsize=fonts)
        plt.xlabel("Tamaño",fontsize=fonts)
        plt.ylabel("Peso",fontsize=fonts)
        plt.scatter(tamaño, peso, c = 'red')
        plt.grid()
        plt.show()
```



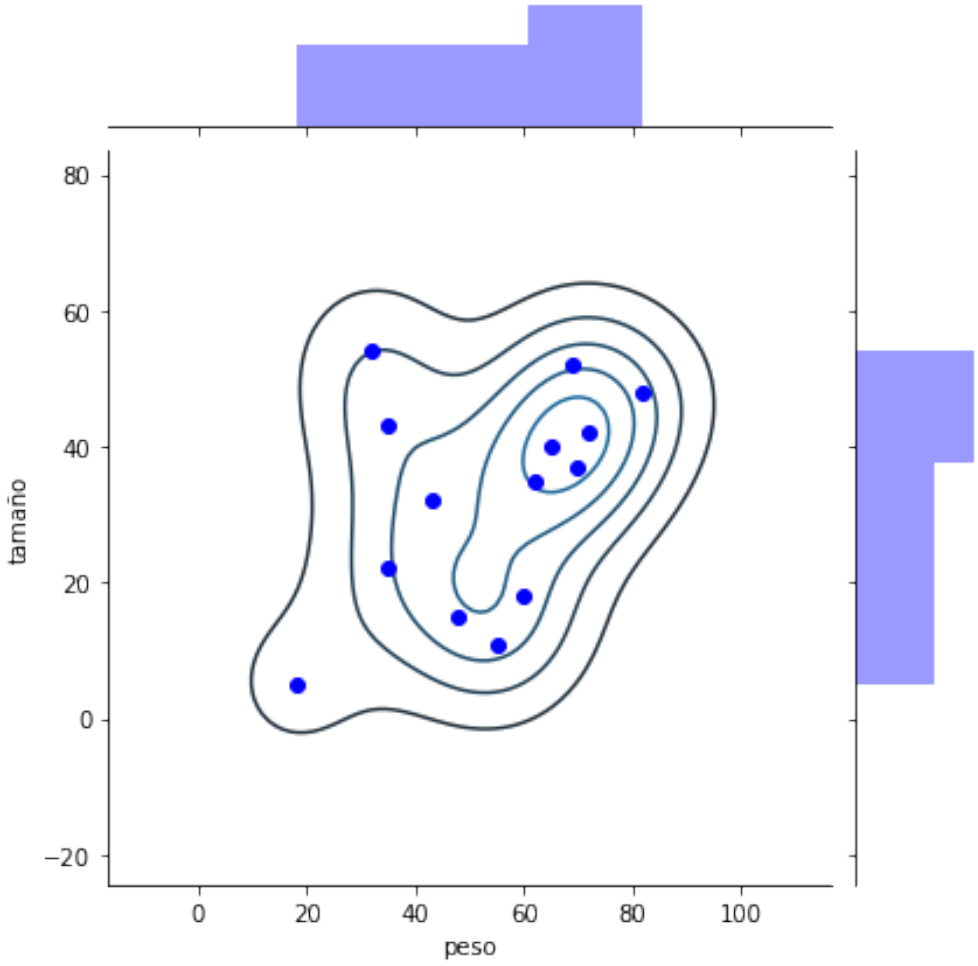
Editar Plot

```
In [5]: plt.figure(figsize=(10,8))
        plt.title("Plot tamaño y peso de ratones",fontsize=fonts)
        plt.xlabel("Tamaño",fontsize=fonts)
        plt.ylabel("Peso",fontsize=fonts)


        plt.yticks(fontsize=fonts)
        plt.scatter(x=tamaño,y=peso,c='darkblue',s=130,edgecolors='k')
        plt.grid (True)
        plt.show()
```



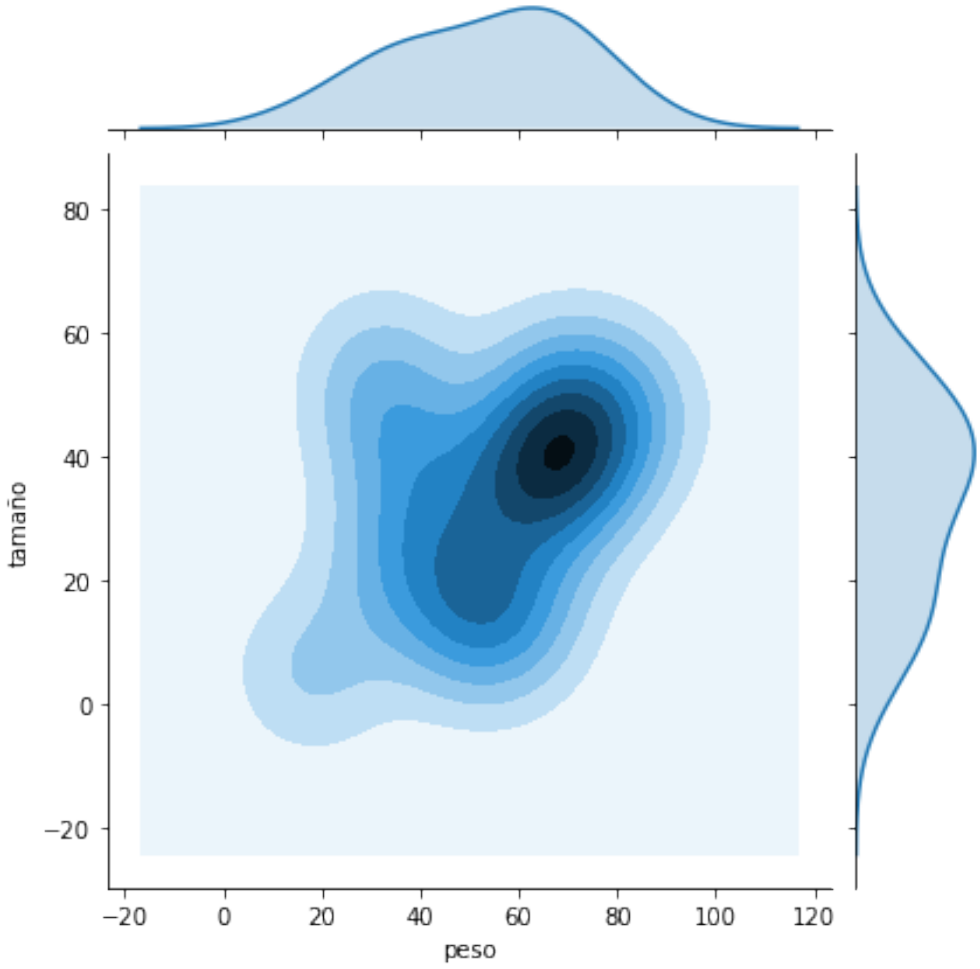
A dataframe

```
In [6]: data = {'peso': peso,
                'tamaño': tamaño}

        df = pd.DataFrame(data)

In [7]: df.head()

Out[7]:    peso  tamaño
        0    55      11
        1    35      22
        2    43      32
        3    62      35
        4    70      37

In [8]: sns.jointplot("peso", "tamaño", data=df, color="b").plot_joint(sns.kdeplot, zorder=0, n_l

Out[8]: <seaborn.axisgrid.JointGrid at 0x7f6b6f4458d0>
```




```
In [9]: sns.jointplot("peso", "tamaño", data=df, kind='kde')
```

```
Out[9]: <seaborn.axisgrid.JointGrid at 0x7f6b6f279f50>
```



Visualizar dataset

- Importar dataset

```
In [10]: df = sns.load_dataset('diamonds') #Importar dataset

In [11]: df
```

Out[11]:

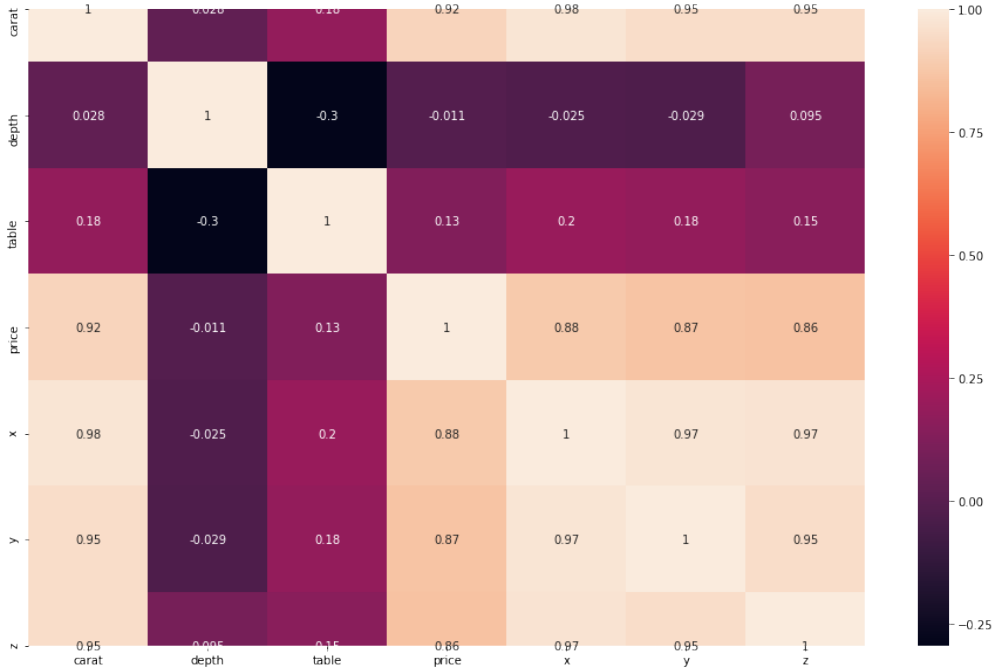
	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

```
...      ...        ...   ...     ...   ...   ...   ...   ...   ...   ...
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 x 10 columns]
```

Correlación de variables

```
In [12]: import numpy as np
         corr = df.corr()
         plt.figure(figsize=(16, 10))
         sns.heatmap(corr, annot=True, fmt='.2g')
```

```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7f6b6f1285d0>
```

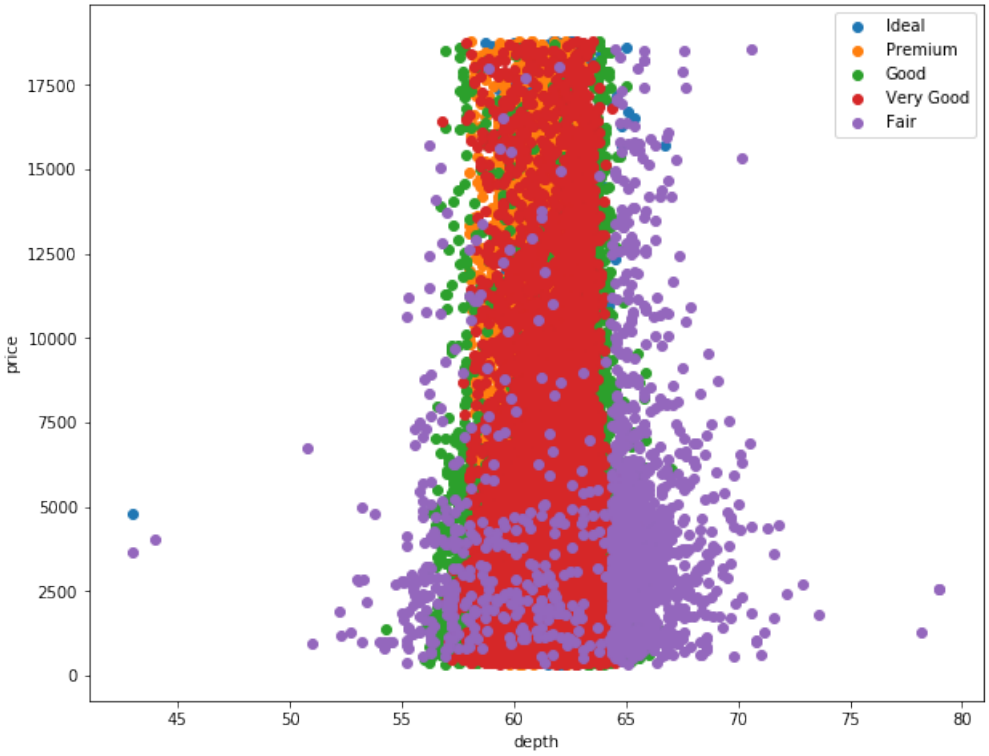


Dos variables por clase

```
In [13]: plt.figure(figsize=(10,8))
         for each in df['cut'].unique():
             subset = df[df['cut'] == each]
             plt.scatter(x=subset['depth'], y=subset['price'], label=each)
         plt.xlabel('depth')
```

```
        plt.ylabel('price')
        plt.legend()
```

Out[13]: <matplotlib.legend.Legend at 0x7f6b6f13fa90>

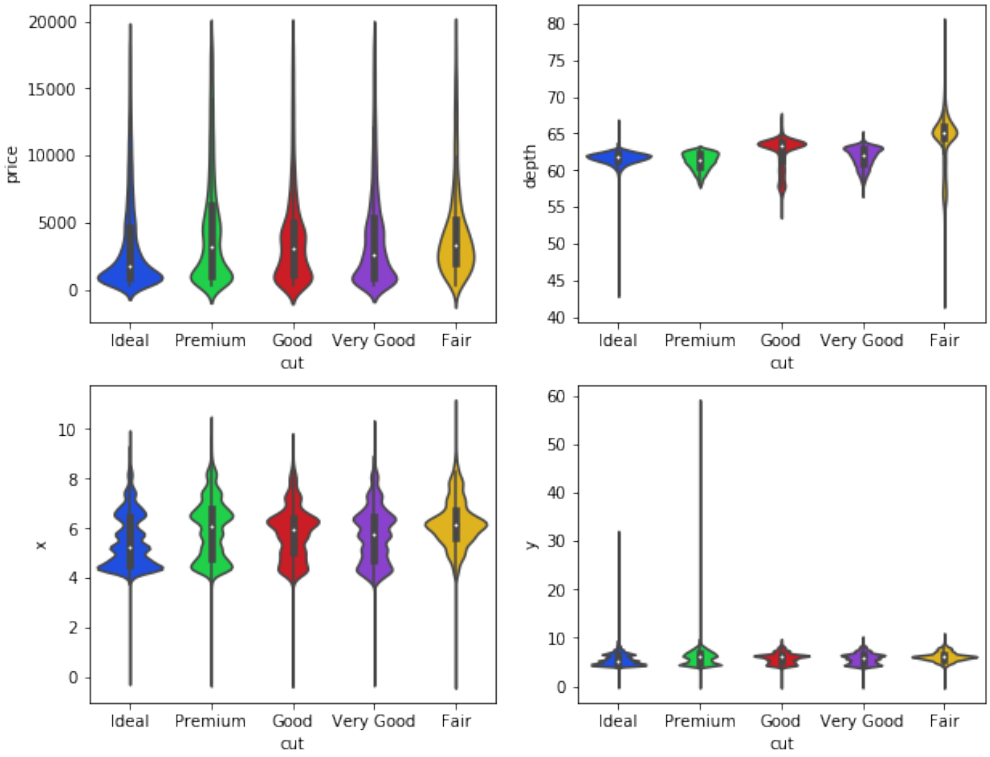


In [14]:
```
fig, ax = plt.subplots()
fig.set_size_inches(10, 8)
sns.violinplot(x=df["cut"], y=df["price"], linewidth=5)
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x7f6b6f35ff50>

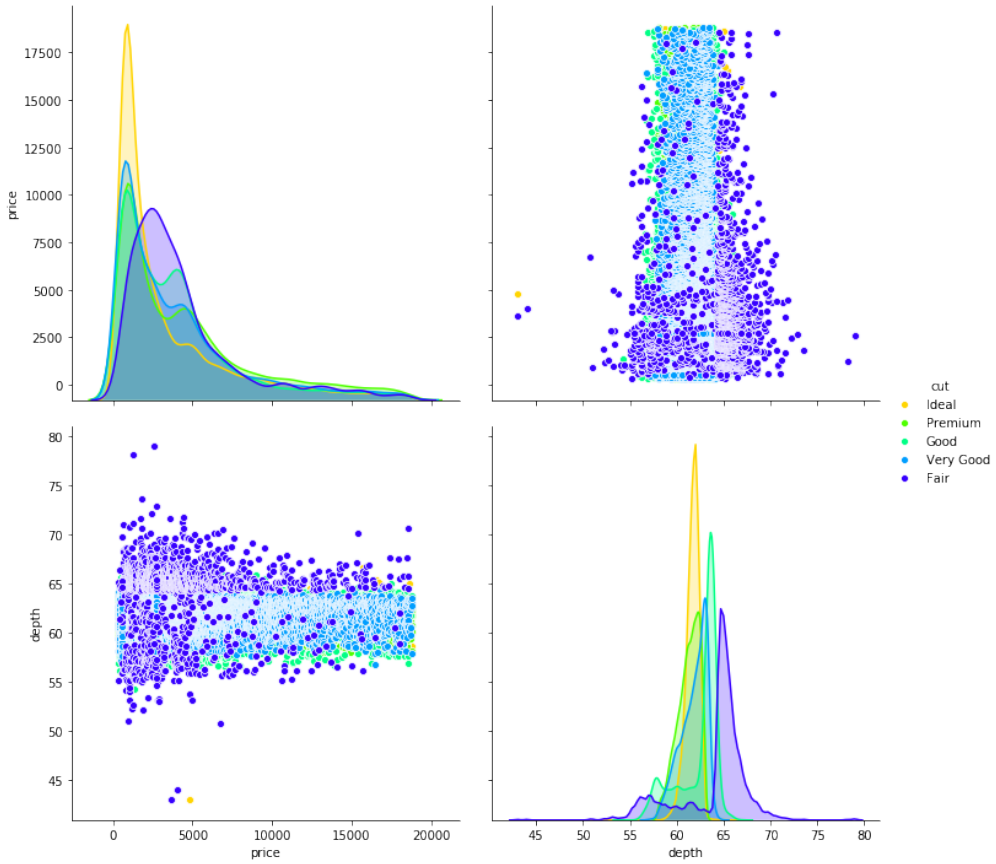
Violinplot de variables por clase

```
In [15]: plt.style.use('seaborn-bright')
         plt.figure(figsize=(10,8))
         plt.subplot(2,2,1)
         sns.violinplot(x = df["cut"], y = df["price"])
         plt.subplot(2,2,2)
         sns.violinplot(x = df["cut"], y = df['depth'])
         plt.subplot(2,2,3)
         sns.violinplot(x = df["cut"], y = df['x'])
         plt.subplot(2,2,4)
         sns.violinplot(x =df["cut"], y = df['y'])

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x7f6b6ea6ed90>
```



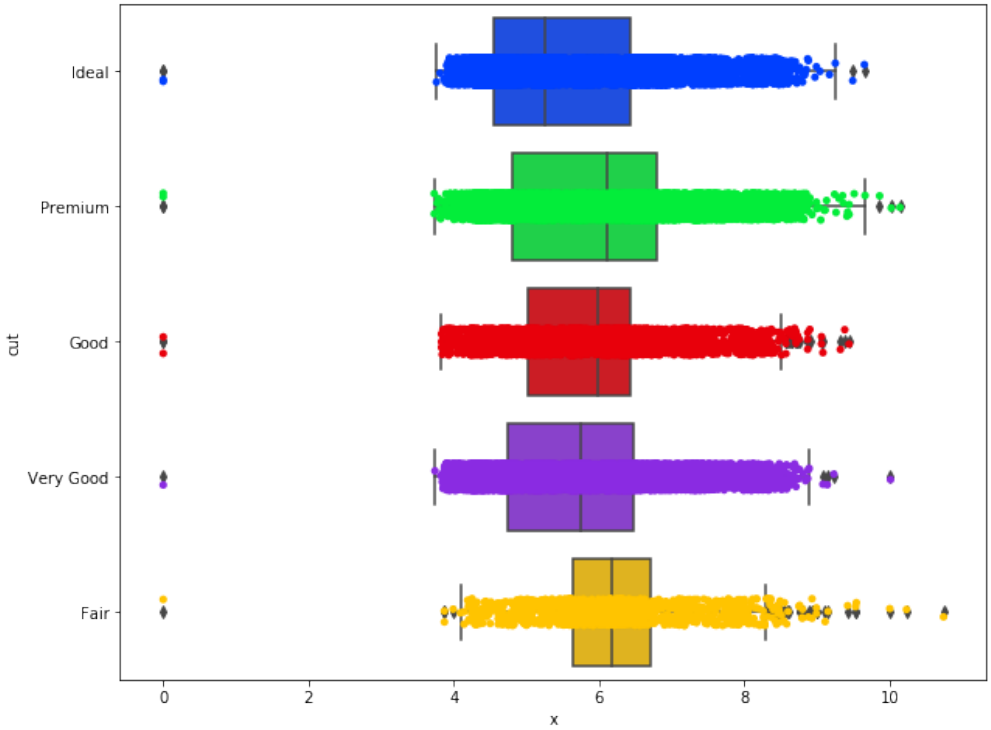
Dos variables y distribución por clase

```
In [16]: sns.pairplot(df, vars=['price', 'depth'], palette=sns.color_palette('hsv'), hue='cut', h
         plt.show()
```



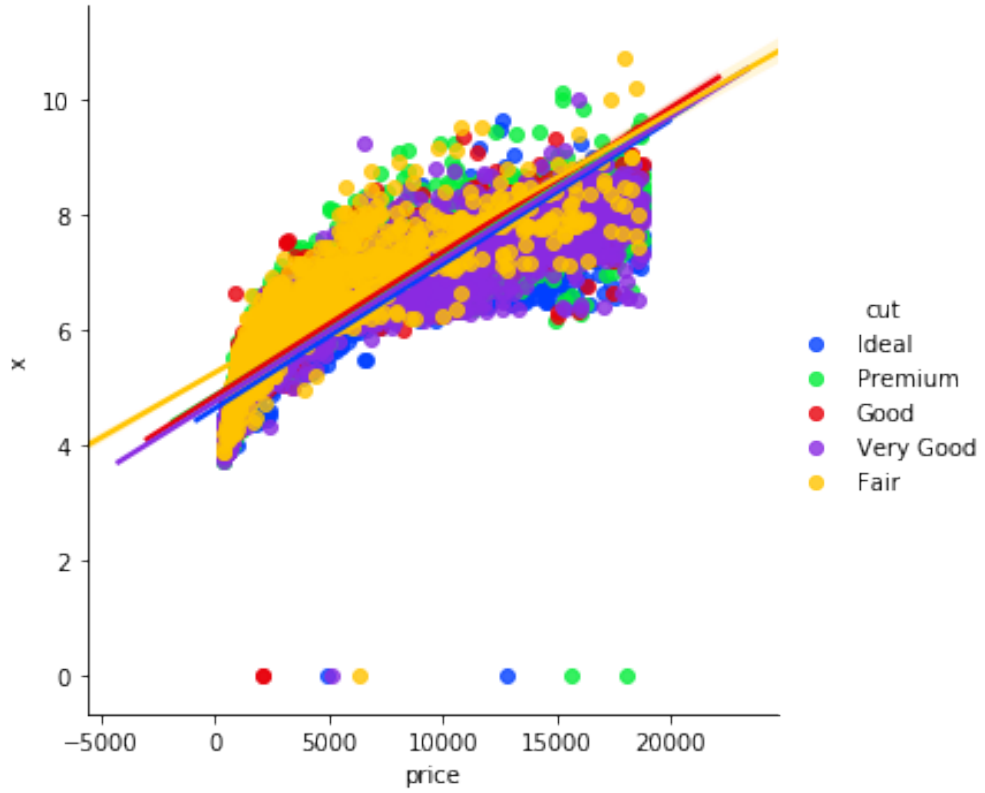
Boxplot de variable por clase

```
In [17]: fig, ax = plt.subplots()
         fig.set_size_inches(10, 8)
         ax = sns.boxplot(data=df, x = 'x',y = 'cut')
         ax = sns.stripplot(data=df, x='x', y='cut', jitter=True, edgecolor='green')
```



Dos variables con lmplot

```
In [18]: sns.lmplot(x='price',y='x',hue='cut',data=df)
         plt.show()
```



- Usar dataset de interés y explorarlo con las herramientas aprendidas en el módulo de preprocesamiento, usar pandas, matplotlib y numpy
- Extraer información y conclusiones
- Explorar otras funciones