El futuro digital es de todos

MinTIC

Análisis y visualización de datos

OPERADO POR:

Universidad Tecnológica de Pereira

Mision TIC 2022

RUTA DE APRENDIZAJE 1

Análisis de datos

Podemos utilizar las funciones de Python para generar graficas y realizar un análisis de datos mas extenso y completo para nuestros problemas.

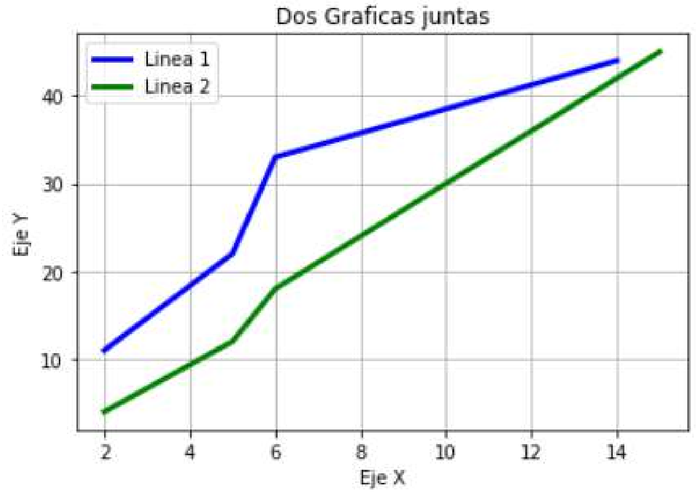
Grafico usando listas:

In [1]:
```
import matplotlib.pyplot as plt
```

In [2]:
```
x1=[2,5,6,14]
y1= [11,22,33,44]

x2=[2,5,6,15]
y2=[4,12,18,45]
```

In [3]:
```
#Graficar 2 lineas en la misma grid
plt.plot(x1,y1, color="blue", linewidth = 3, label = 'Linea 1')
plt.plot(x2,y2, color="green", linewidth = 3, label = 'Linea 2')

plt.title('Dos Graficas juntas')
plt.xlabel('Eje X')
plt.ylabel('Eje Y')
plt.legend()
plt.grid()
plt.show()
```

In [3]:
```
#Graficar 2 lineas en la misma grid
plt.plot(x1,y1, color="blue", linewidth = 3, label = 'Linea 1')
plt.plot(x2,y2, color="green", linewidth = 3, label = 'Linea 2')

plt.title('Dos Graficas juntas')
plt.xlabel('Eje X')
plt.ylabel('Eje Y')
plt.legend()
plt.grid()
plt.show()
```

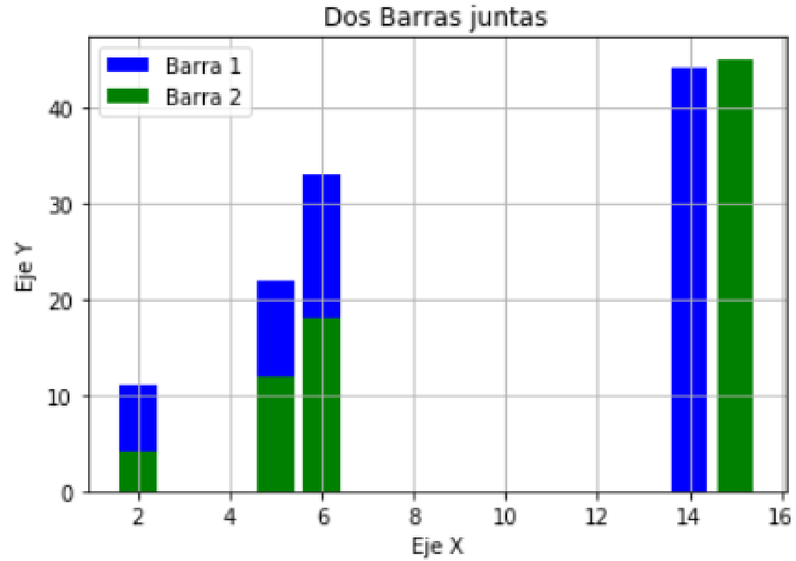


Python es una herramienta muy fuerte para ayudarnos en la resolución de problemas y especialmente para el análisis de datos, permitiendo visualizar la información o los resultados de una forma mas eficiente y fácil de manejar por medio de grafica de datos.

También podemos utilizar **plt.bar()** para generar una grafica de barras, donde necesitaremos especificar los datos de cada barra, su color y su nombre.

Es posible desarrollar diferentes tipos de graficas para realizar un análisis más detallados de resultados. Uno de los mas comunes es el **grafico de barras**, donde cada barra reasenta un valor numero y permite visualizar características del problema de forma más rápida.

```
In [4]: #Grafico de barras. Cuando coinciden la misma X, las apila
plt.bar(x1,y1, color="blue", linewidth = 3, label = 'Barra 1')
plt.bar(x2,y2, color="green", linewidth = 3, label = 'Barra 2')

plt.title('Dos Barras juntas')
plt.xlabel('Eje X')
plt.ylabel('Eje Y')
plt.legend()
plt.grid()
plt.show()
```

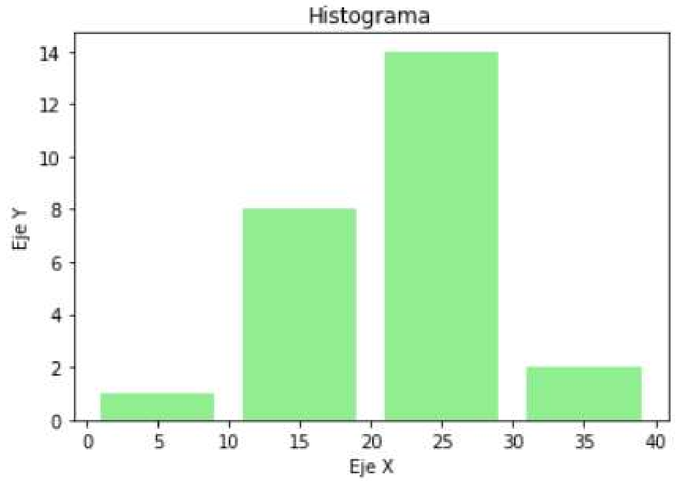


Podemos generar histogramas con la función **plt.hist(),** debemos indicar nuevamente que d atos utilizaremos para hacer el análisis de la i nformación, su rango, color y ancho de barra.

Un histograma simboliza la distribución de un conjunto de datos. Sirven para obtener una "primera vista" general, o panorama, de la dist ribución de la población, o de la muestra, resp ecto a una característica, cuantitativa y contin ua.

In [5]:
```
#Histogramas
Datos = [20,22,21,20,23,25,28,40,22,23,22,15,16,18,18,19,21,22,24,4,12,17,17,22,30,]
Rangobin=[0,10,20,20,30,40]
```

In [6]:
```
plt.hist(Datos, Rangobin, histtype='bar',rwidth=0.8, color='lightgreen')

plt.title('Histograma')
plt.xlabel('Eje X')
plt.ylabel('Eje Y')
#plt.grid()  #la grid no es necesaria (lineas horizontales y verticales)
plt.show()
```



In [8]:
```
#Pie
valores =[20,40,60,80]

plt.pie(valores, labels=['Prekinder', 'kinder', 'primaria', 'secundaria'], colors=['red','purple','blue','orange']
        , startangle=90,shadow=True, explode=(0.1,0,0,0),autopct='%1.1f%%')
plt.title('Grafico circular')
plt.show()
```

Ejemplo: Tenemos los resultados de la información recolectada sobre la cantidad de niños en comparación a la cantidad de mascotas que se encuentran en 3 ciudades principales del país.

Numéricamente podemos identificar si existen mas o menos mascotas en la ciudad, sin embargo podemos apoyarnos de las herramientas que hemos revisado podemos generar graficas que nos permitan visualizar los resultados de una forma diferente y dar conclusiones mas rápidamente y con seguridad.

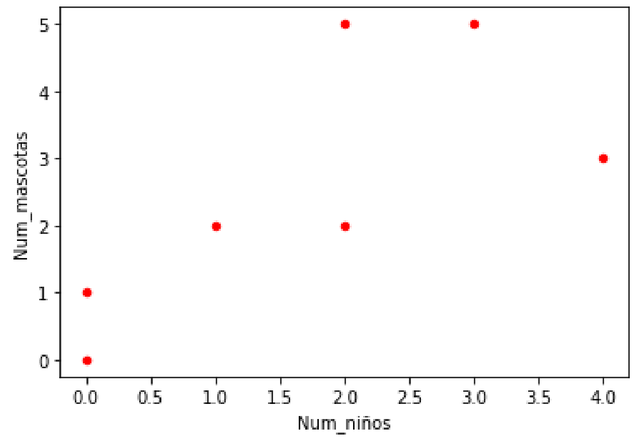
In [39]:
```
import pandas as pd

df = pd.DataFrame({
    'Nombre':['john','maria','pedro','jenifer','bob','lisa','jose'],
    'Edad':[23,78,22,19,45,33,20],
    'Ciudad':['Bogota','Medellin','Bogota','Medellin','Bogota','Armenia','Armenia'],
    'Num_niños':[2,0,0,3,2,1,4],
    'Num_mascotas':[5,1,0,5,2,2,3]
})
```

In [40]:
```
df.head()
```

Out[40]:

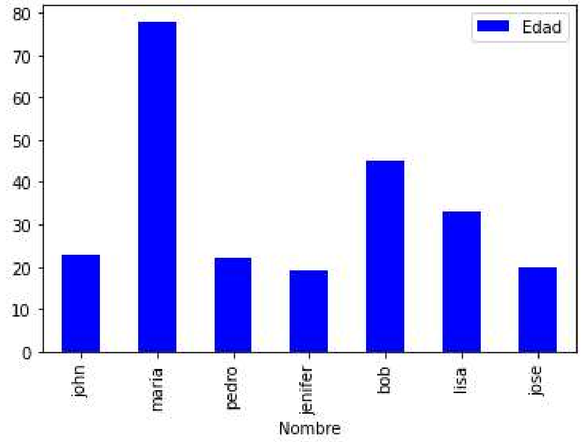
	Nombre	Edad	Ciudad	Num_niños	Num_mascotas
0	john	23	Bogota	2	5
1	maria	78	Medellin	0	1
2	pedro	22	Bogota	0	0
3	jenifer	19	Medellin	3	5
4	bob	45	Bogota	2	2

In [41]:
```
df.plot(kind='scatter',x='Num_niños',y='Num_mascotas',color='red')
plt.show()
```

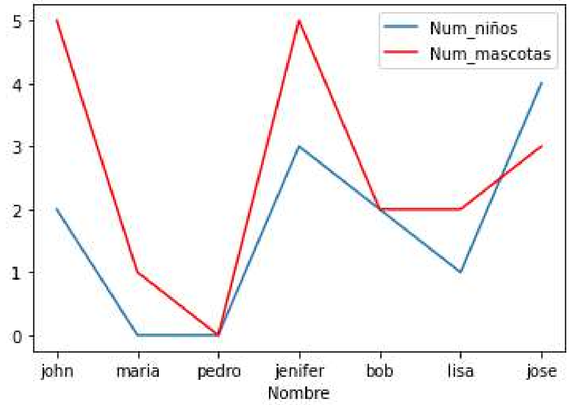


In [42]:
```
df.plot(kind='bar',x='Nombre',y='Edad', color='blue')
```

Out[42]: <AxesSubplot:xlabel='Nombre'>



In [43]:
```
# gca stands for 'get current axis'
ax = plt.gca()

df.plot(kind='line',x='Nombre',y='Num_niños',ax=ax)
df.plot(kind='line',x='Nombre',y='Num_mascotas', color='red', ax=ax)

plt.show()
```

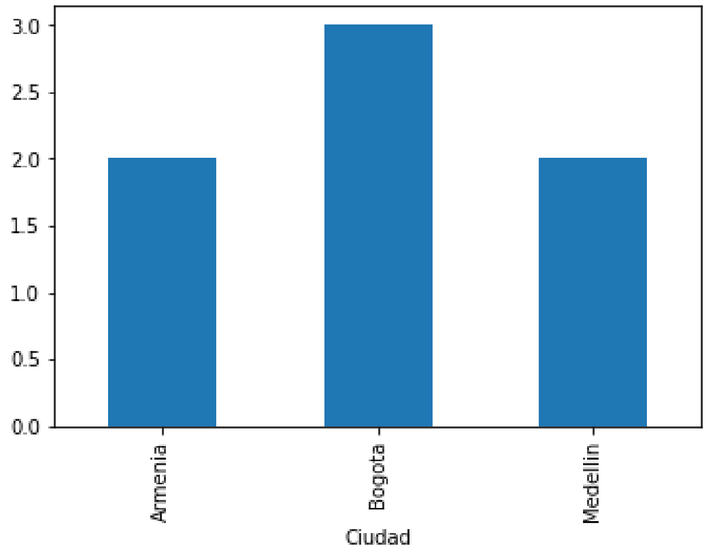


Groupby-Series:

El método **pandas.DataFrame.groupby** tiene una funcionalidad semejante a la vista para series, con los condicionantes propios de los **dataframes**: es necesario indicar el eje que contiene el criterio por el que se va a realizar la agrupación.

```
In [47]: #Serie cuyo indice es la ciudad, y valor es la cantidad de unicos
         print(type(df.groupby('Ciudad')['Nombre'].nunique()))
         df.groupby('Ciudad')['Nombre'].nunique().plot(kind='bar')
         plt.show()
```

```
<class 'pandas.core.series.Series'>
```

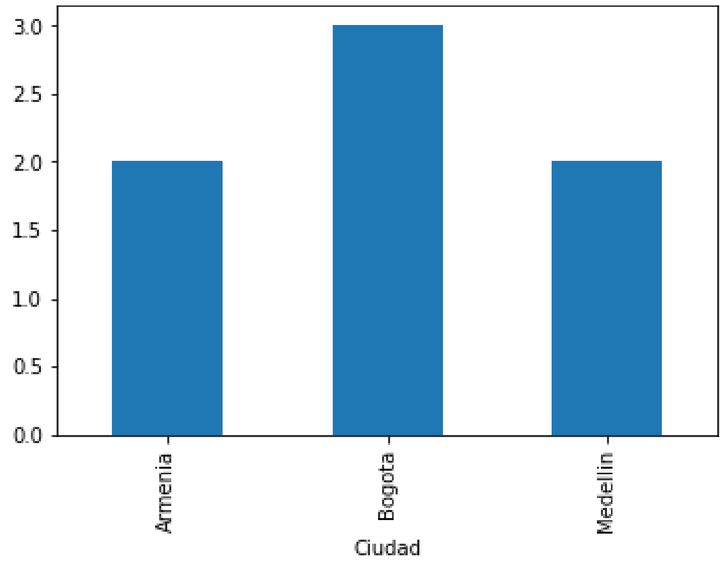


Podemos de igual manera mostrar el tipo directamente imprimiendo la información. Y utilizar la función para agrupar los datos y genera runa lectura mejorada de la base de datos o los resultados obtenidos.

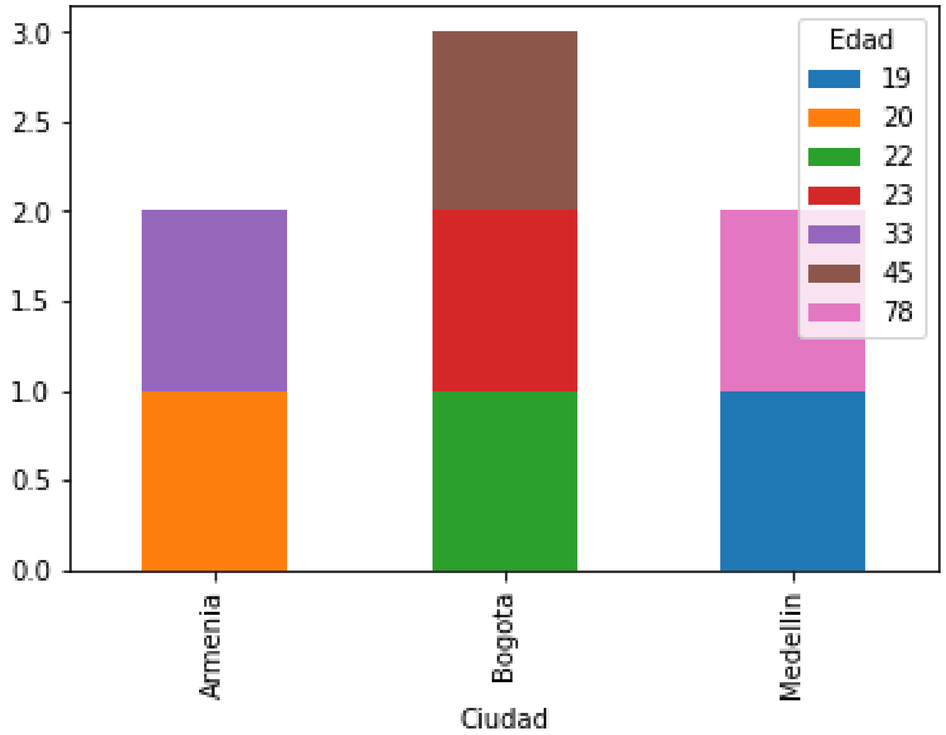
```
In [52]: print(type(df['Ciudad']))
         print(type(df[['Ciudad','Edad']]))
         print(type(df.groupby('Ciudad')))
         print(type(df.groupby('Ciudad')['Nombre']))
         print(type(df.groupby('Ciudad')['Nombre'].nunique()))
```

```
<class 'pandas.core.series.Series'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
<class 'pandas.core.groupby.generic.SeriesGroupBy'>
<class 'pandas.core.series.Series'>
```

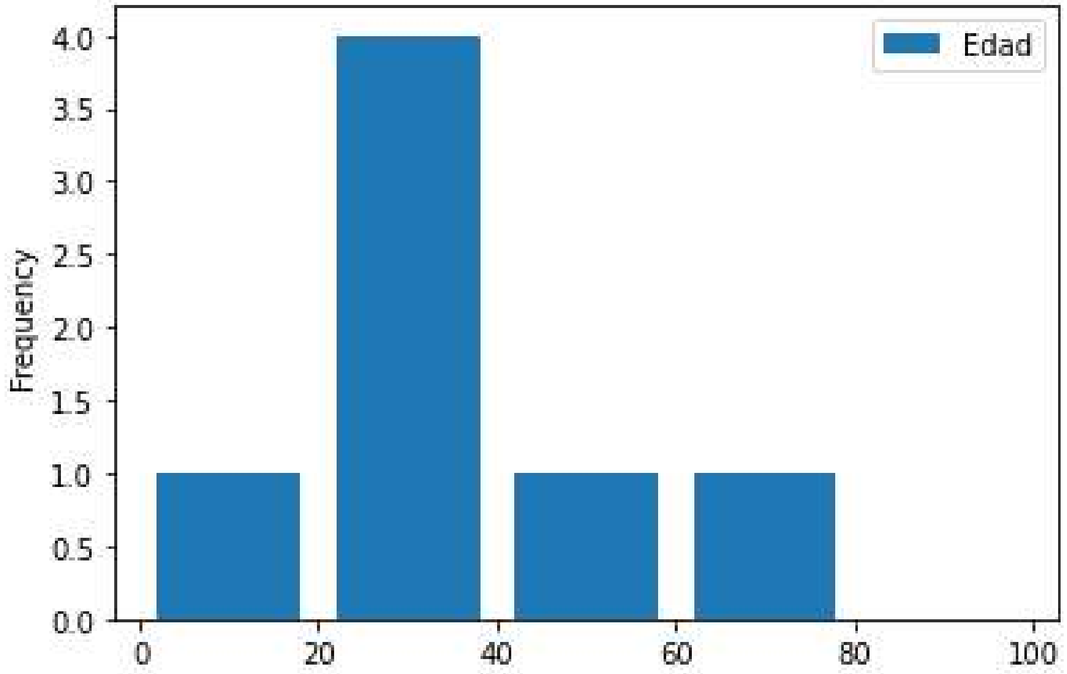
```
In [53]: df.groupby('Ciudad')['Nombre'].nunique().plot(kind='bar')
         plt.show()
```



```
In [58]: df.groupby(['Ciudad','Edad']).size().unstack().plot(kind='bar',stacked=True)
         plt.show()
```



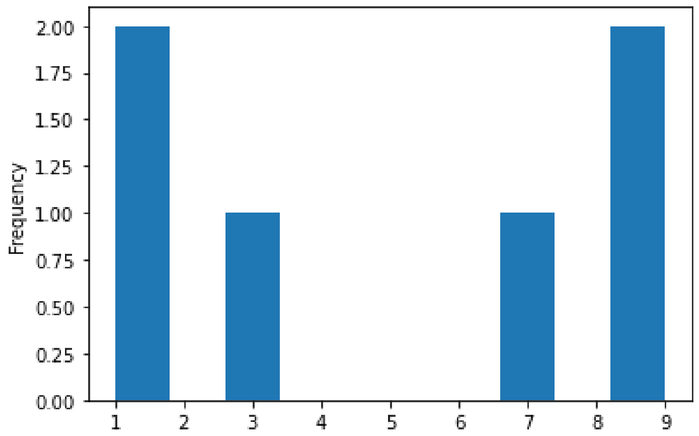
```
In [61]: df[['Edad']].plot(kind='hist',bins=[0,20,40,60,80,100],rwidth=0.8)
         plt.show()
```



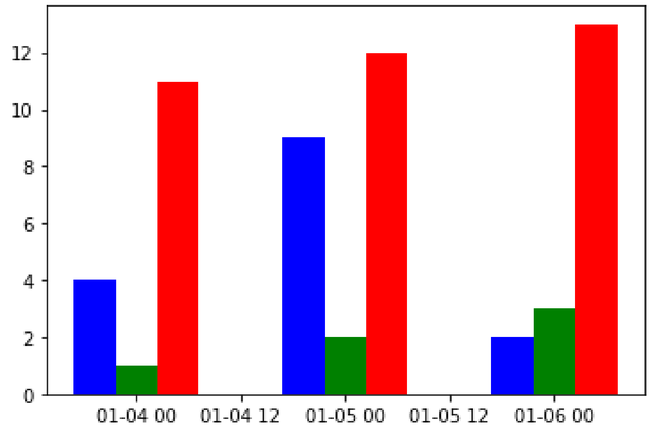
Dataframe con campo tipo fecha:

In [30]:
```
import pandas as pd
import matplotlib.pyplot as plt

# source dataframe using an arbitrary date format (m/d/y)
df = pd.DataFrame({
    'name':[
        'john','lisa','peter','carl','linda','betty'
    ],
    'date_of_birth':[
        '01/21/1988','03/10/1977','07/25/1999','01/22/1977','09/30/1968','09/15/1970'
    ]
})
```

In [31]:
```
df['date_of_birth'] = pd.to_datetime(df['date_of_birth'], infer_datetime_format=True)

plt.clf()
df['date_of_birth'].map(lambda d: d.month).plot(kind='hist')
plt.show()
```

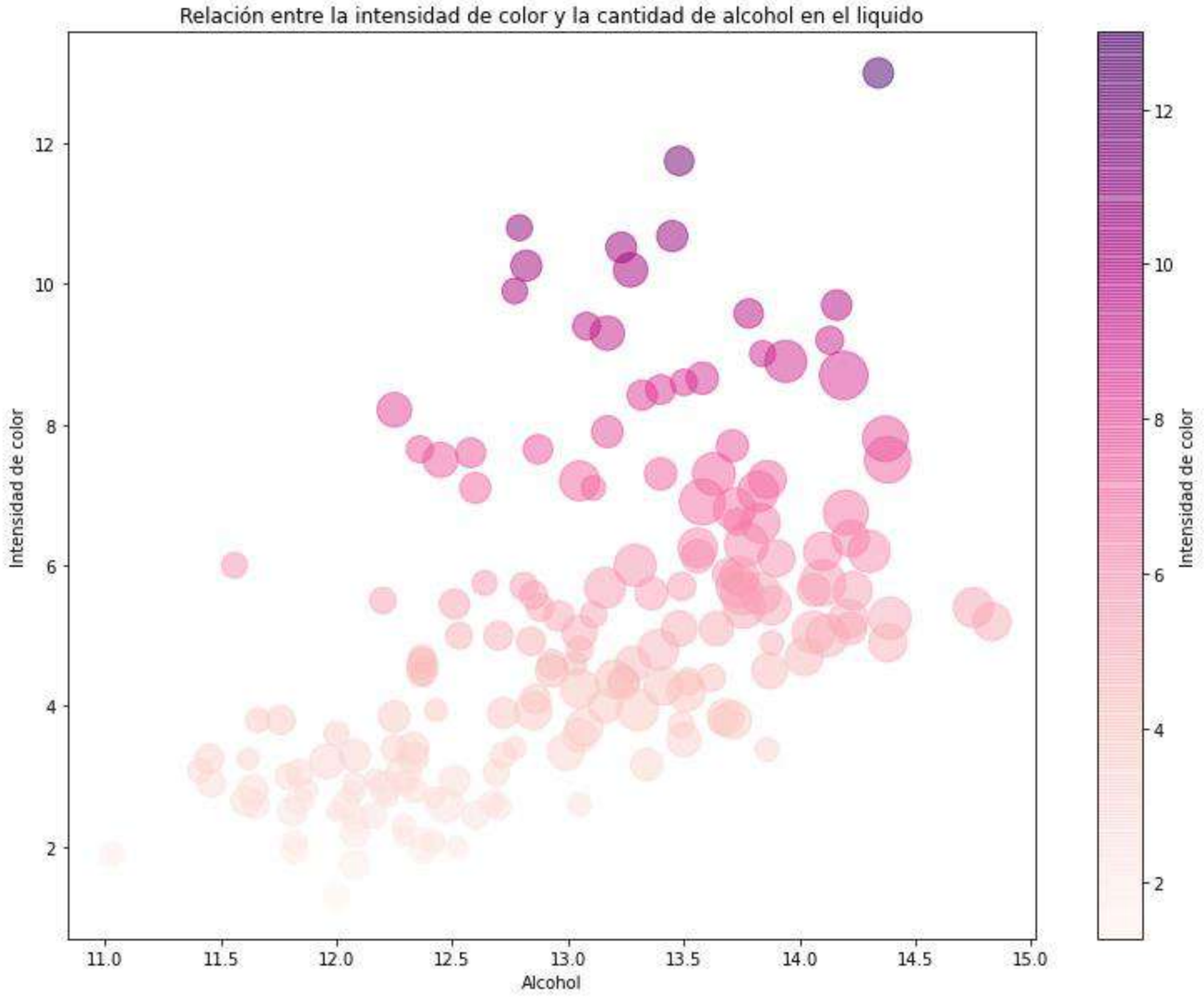


In [32]:
```
import matplotlib.pyplot as plt
from matplotlib.dates import date2num
import datetime

x = [
    datetime.datetime(2011, 1, 4, 0, 0),
    datetime.datetime(2011, 1, 5, 0, 0),
    datetime.datetime(2011, 1, 6, 0, 0)
]
x = date2num(x)

y = [4, 9, 2]
z = [1, 2, 3]
k = [11, 12, 13]

ax = plt.subplot(111)
ax.bar(x-0.2, y, width=0.2, color='b', align='center')
ax.bar(x, z, width=0.2, color='g', align='center')
ax.bar(x+0.2, k, width=0.2, color='r', align='center')
ax.xaxis_date()

plt.show()
```



Con Python tenemos miles de posibilidades para visualizar nuestra información, podemos jugar con los colores y su intensidad para realizar graficas mas detalladas o mas fáciles de leer:

In [62]:
```
import pandas as pd
import matplotlib.pyplot as plt

#import data and create dataframe
liquido_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data'
liquido_column_headers = ['Alcohol','Acido Malico','Cenizas','Alcalinidad de las cenizas',
                          'Magnesio','Fenoles totales','Flavonoides',
                          'Fenoles no flavonoides','Proantocianinas','Intensidad de color',
                          'Matiz','OD280/OD315 de sustancia diluida' ,'Prolina']
liquido_df = pd.read_csv(liquido_url, names = liquido_column_headers)

#figure
fig, ax1 = plt.subplots()
fig.set_size_inches(13, 10)

#labels
ax1.set_xlabel('Alcohol')
ax1.set_ylabel('Intensidad de color')
ax1.set_title('Relación entre la intensidad de color y la cantidad de alcohol en el liquido')

#c sequence
c = liquido_df['Intensidad de color']

#plot
plt.scatter( liquido_df['Alcohol'], liquido_df['Intensidad de color'] , c=c,
            cmap = 'RdPu', s = liquido_df['Prolina']*.5, alpha =0.5)
cbar = plt.colorbar()
cbar.set_label('Intensidad de color')
```


Relación entre la intensidad de color y la cantidad de alcohol en el liquido
Intensidad de color
Alcohol
11.0
11.5
12.0
12.5
13.0
13.5
14.0
14.5
15.0
2
4
6
8
10
12
Intensidad de color

El futuro digital es de todos

MinTIC

GRACIAS

OPERADO POR:

Universidad Tecnológica de Pereira

MisiónTIC2022