ITS INFORMATION AND COMMUNICATIONS TECHNOLOGY Academy

NOME MODULO: *IA*

UNITÀ DIDATTICA: IA.1

Lezione 2

Stefano Puglia

Biennio 2024-2026

matplotlib

INDICE DEGLI ARGOMENTI

- Line plot
- Bar chart
- Pie chart

its ICTAcademy
INFORMATION AND COMMUNICATIONS TECHNOLOGY

Line plot

- Tipo di plot che mostra informazioni come una serie di “punti dato” chiamati “marker” connessi da segmenti rette (linea spezzata)
- Use case: dataset “continuo” lungo un periodo di tempo → studio di trend
- pandas DataFrame e Series usano direttamente il metodo **plot**
- In una Series, il metodo plot → x=index, y=column
- In un DataFrame il metodo plot → richiede “trasposta”
- Matplotlib pyplot entra in gioco per maggiori capacità

Line plot

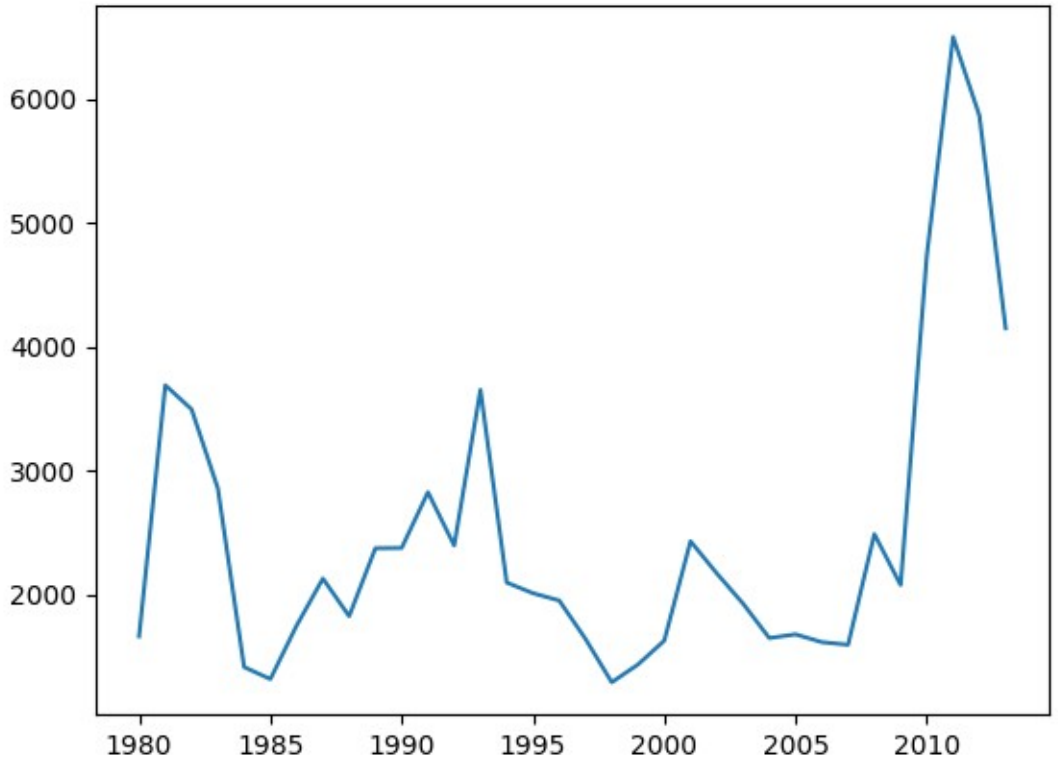
```
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
haiti = df.loc['Haiti', years]
print(haiti.head())
```

```
1980    1666
1981    3692
1982    3498
1983    2860
1984    1418
Name: Haiti, dtype: object
```

La pandas **Series** viene automaticamente creata con gli anni come indice ed una colonna di valori corrispondenti agli anni (i.e. valori di ciascun anno)

Line plot

```
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
haiti = df.loc['Haiti', years]
haiti.plot()
```



API: pandas.DataFrame.plot

Line plot

```
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
china_india = df.loc[['China','India'], years]
print(china_india)
```

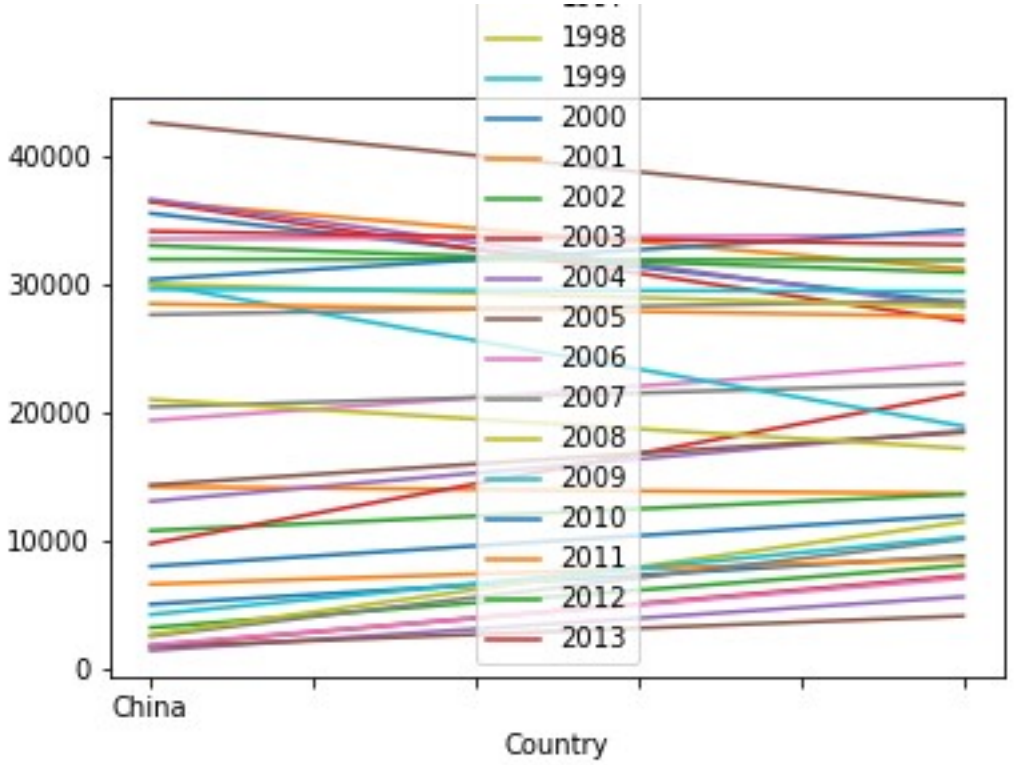
	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
India	8880	8670	8147	7338	5704	4211	7150	10189	11522	10343	...	28235	36210	33848	28742	28261	29456	34235	27509	30933	33087
China	5123	6682	3308	1863	1527	1816	1960	2643	2758	4323	...	36619	42584	33518	27642	30037	29622	30391	28502	33024	34129

2 rows × 34 columns

Il pandas **DataFrame** viene automaticamente creato con i paesi come righe e tante colonne di valori corrispondenti agli anni (i.e. valori di ciascun anno)

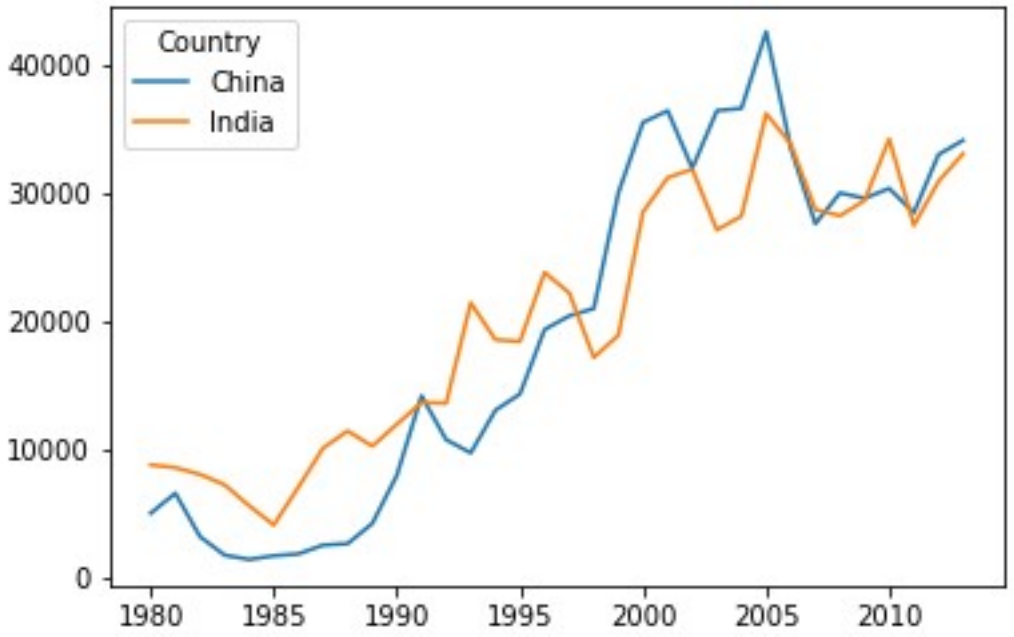
Line plot

```
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
china_india = df.loc[['China','India'], years]
china_india.plot()
```



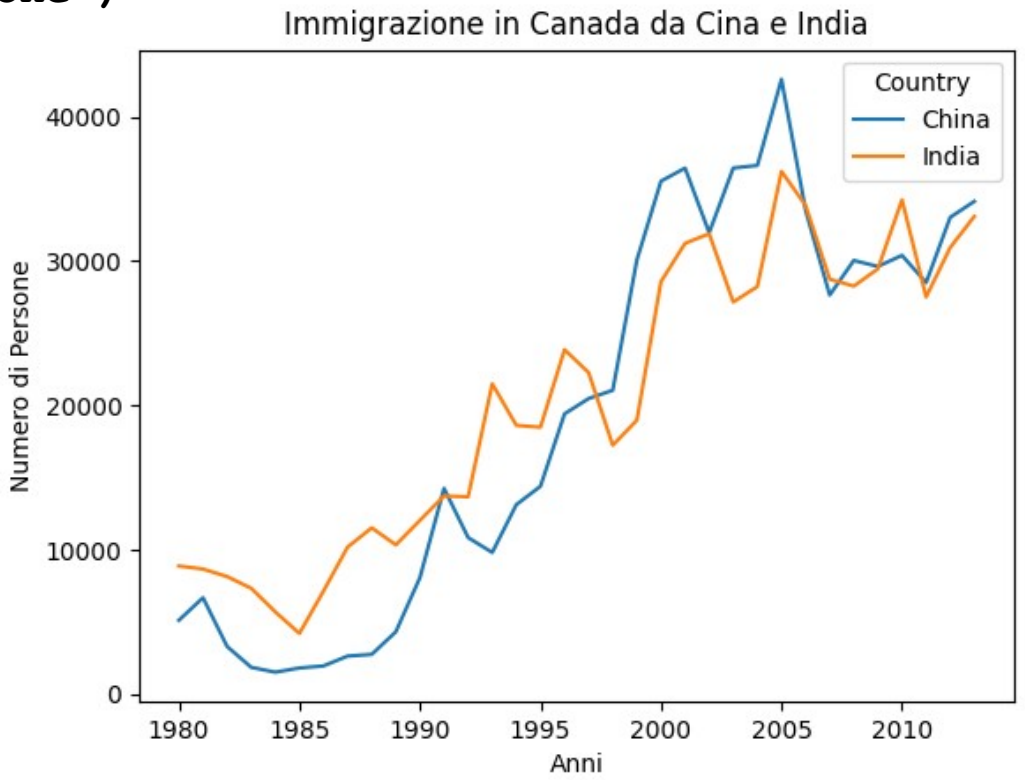
Line plot

```
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
china_india = df.loc[['China','India'], years]
china_india.transpose().plot()
```



Line plot

```
import matplotlib.pyplot as plt
years = list(map(str, range(1980, 2014)))
df.index = df['Country']
china_india = df.loc[['China','India'], years]
china_india.transpose().plot(kind="line")
plt.title('Immigrazione in Canada da Cina e India')
plt.ylabel('Numero di Persone')
plt.xlabel('Anni')
plt.show()
```

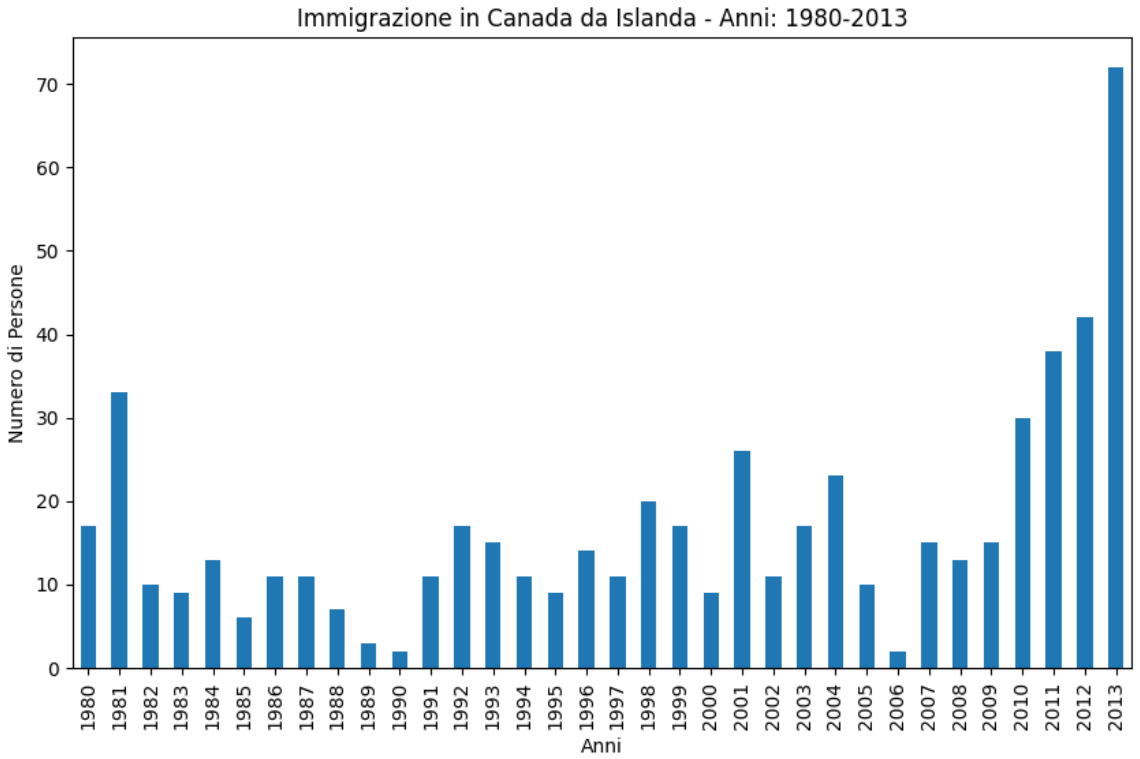


Bar chart

- Tipo di plot in cui la lunghezza delle “barre” rappresenta la grandezza/dimensione della variabile/colonna/caratteristica
- Use case: studio comparativo dei valori di una variabile ad un certo momento
- Esempio: 1) i 10 prodotti più venduti da un esercizio commerciale in un anno; 2) prezzo massimo auto a seconda della marca; 3) media contagi/popolazione di regioni geografiche
- Verticale/orizzontale
- Colori e annotazioni

Bar chart

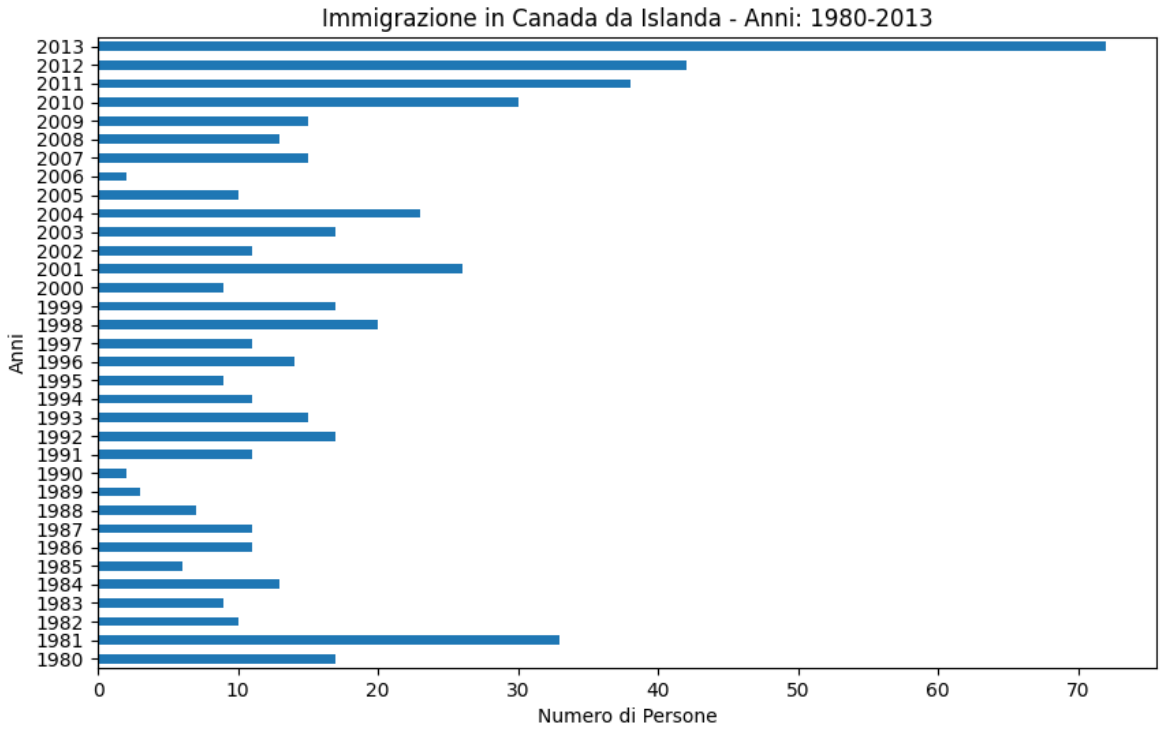
```
df.index = df['Country']
iceland = df.loc['Iceland', self.years]
iceland.plot(kind='bar', figsize=(10, 6))
plt.title('Immigrazione in Canada da Islanda - Anni: 1980-2013')
plt.ylabel('Numero di Persone')
plt.xlabel('Anni')
plt.show()
```



API: pandas.DataFrame.plot

Bar chart

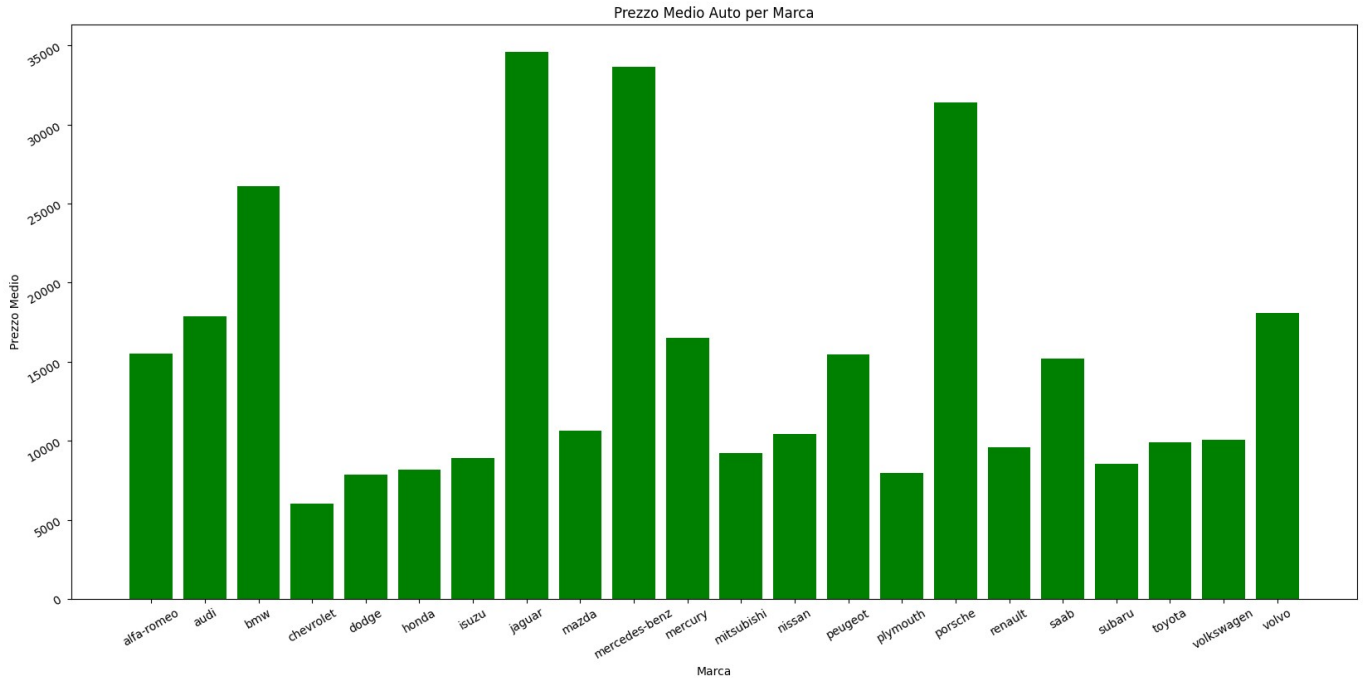
```
df.index = df['Country']
iceland = df.loc['Iceland', self.years]
iceland.plot(kind='barh', figsize=(10, 6))
plt.title('Immigrazione in Canada da Islanda - Anni: 1980-2013')
plt.ylabel('Anni')
plt.xlabel('Numero di Persone')
plt.show()
```



API: pandas.DataFrame.plot

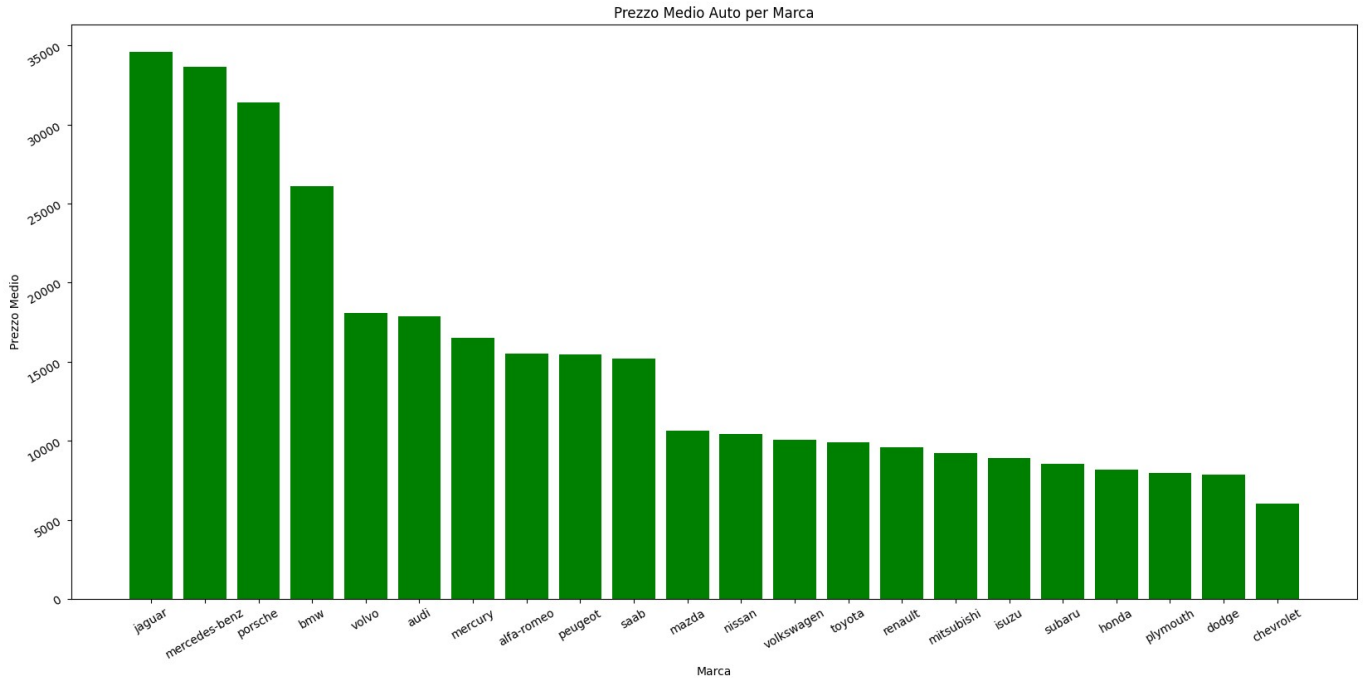
Bar chart

```
df_agg = df.groupby('make').agg(mean_price=('price', 'mean'))
plt.figure(figsize=(20, 9))
plt.bar(x=df_agg.index.astype(str), height=df_agg['mean_price'], color='green')
plt.title('Prezzo Medio Auto per Marca')
plt.ylabel('Prezzo Medio')
plt.xlabel('Marca')
plt.xticks(rotation=30)
plt.yticks(rotation=30)
plt.show()
```



Bar chart

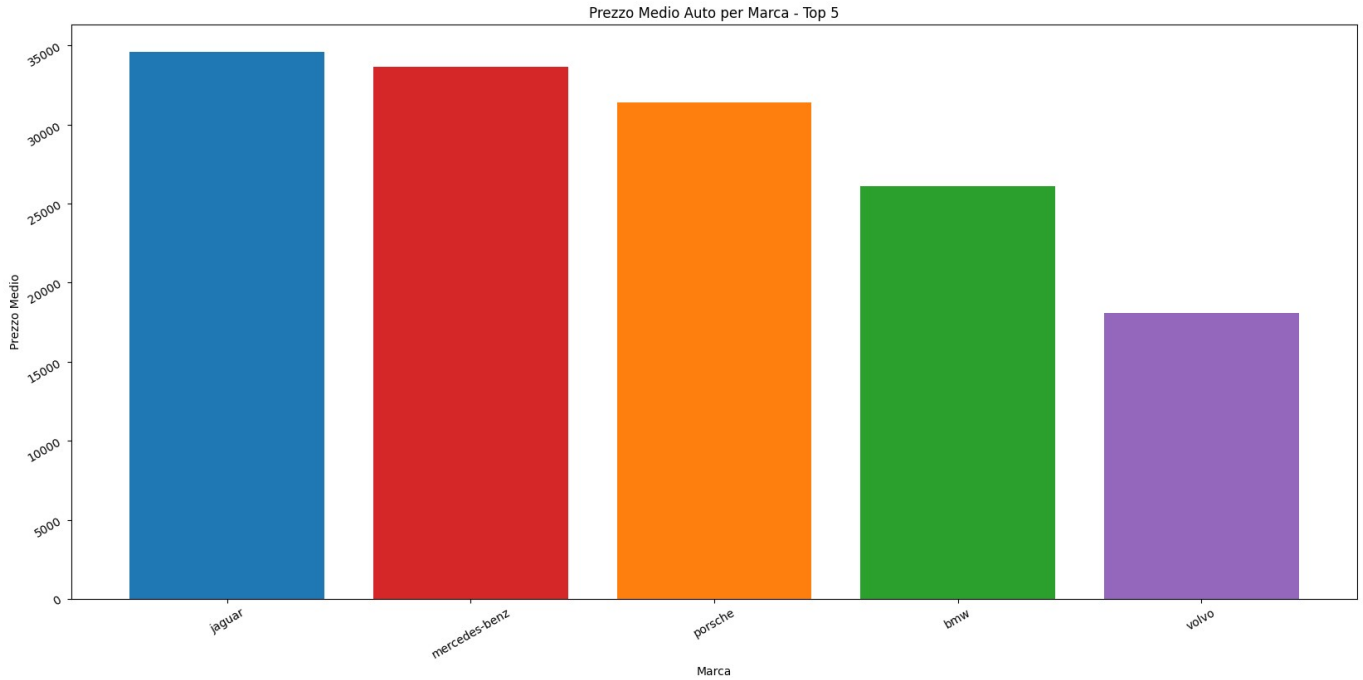
```
df_agg = df.groupby('make').agg(mean_price=('price', 'mean'))
df_agg_mean_down = df_agg.sort_values(by='mean_price', ascending=False)
plt.figure(figsize=(20, 9))
plt.bar(x=df_agg_mean_down.index.astype(str),
        height=df_agg_mean_down['mean_price'], color='green')
plt.title('Prezzo Medio Auto per Marca')
plt.ylabel('Prezzo Medio')
plt.xlabel('Marca')
plt.xticks(rotation=30)
plt.yticks(rotation=30)
plt.show()
```



ICTAcademy
INFORMATION AND COMMUNICATIONS TECHNOLOGY

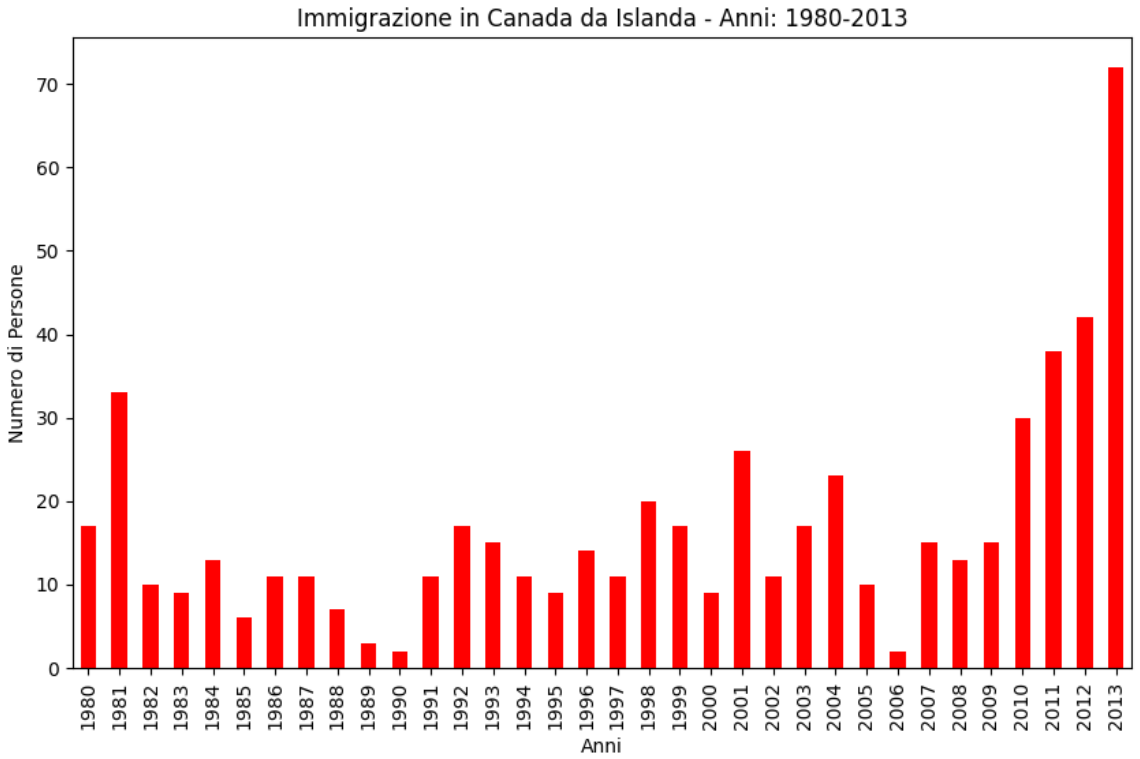
Bar chart

```
...
bar_colors = ['tab:blue', 'tab:red', 'tab:orange', 'tab:green', 'tab:purple']
plt.bar(x=df_agg_mean_down.index.head().astype(str),
        height=df_agg_mean_down['mean_price'].head(),
        color=bar_colors)
...
```



Bar chart

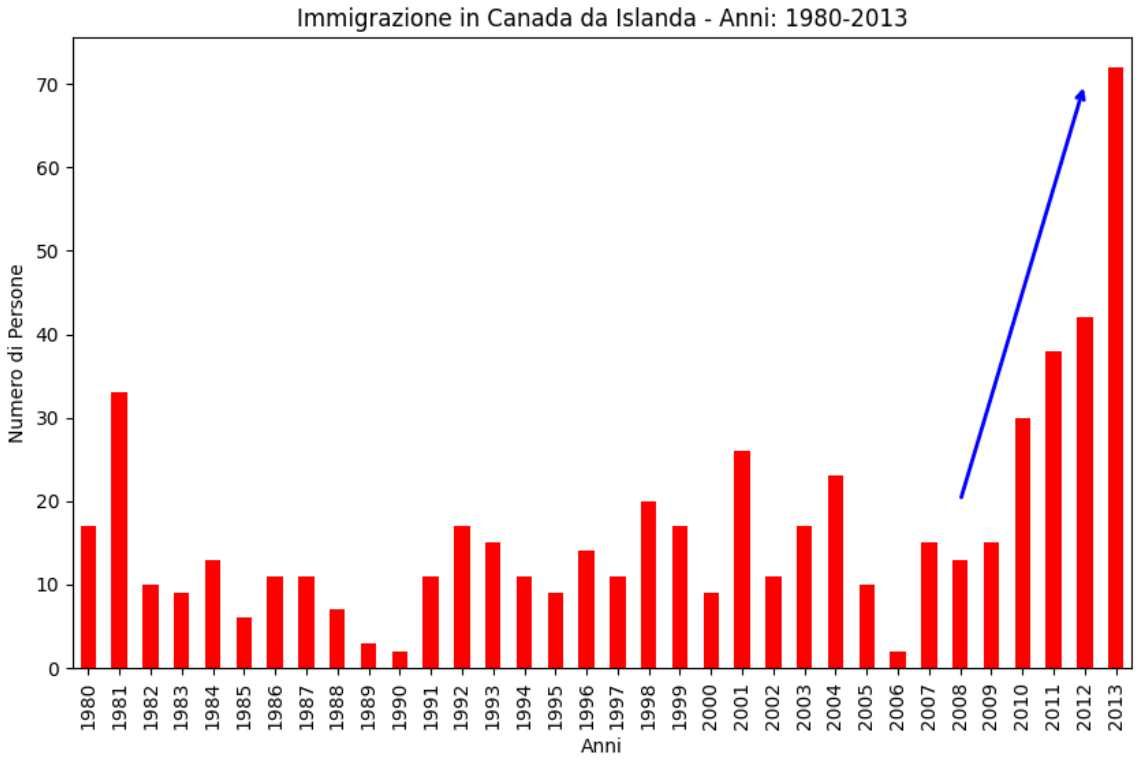
```
df.index = df['Country']
iceland = df.loc['Iceland', self.years]
iceland.plot(kind='bar', figsize=(10, 6), color='red')
plt.title('Immigrazione in Canada da Islanda - Anni: 1980-2013')
plt.ylabel('Numero di Persone')
plt.xlabel('Anni')
plt.show()
```



ICTAcademy
INFORMATION AND COMMUNICATIONS TECHNOLOGY

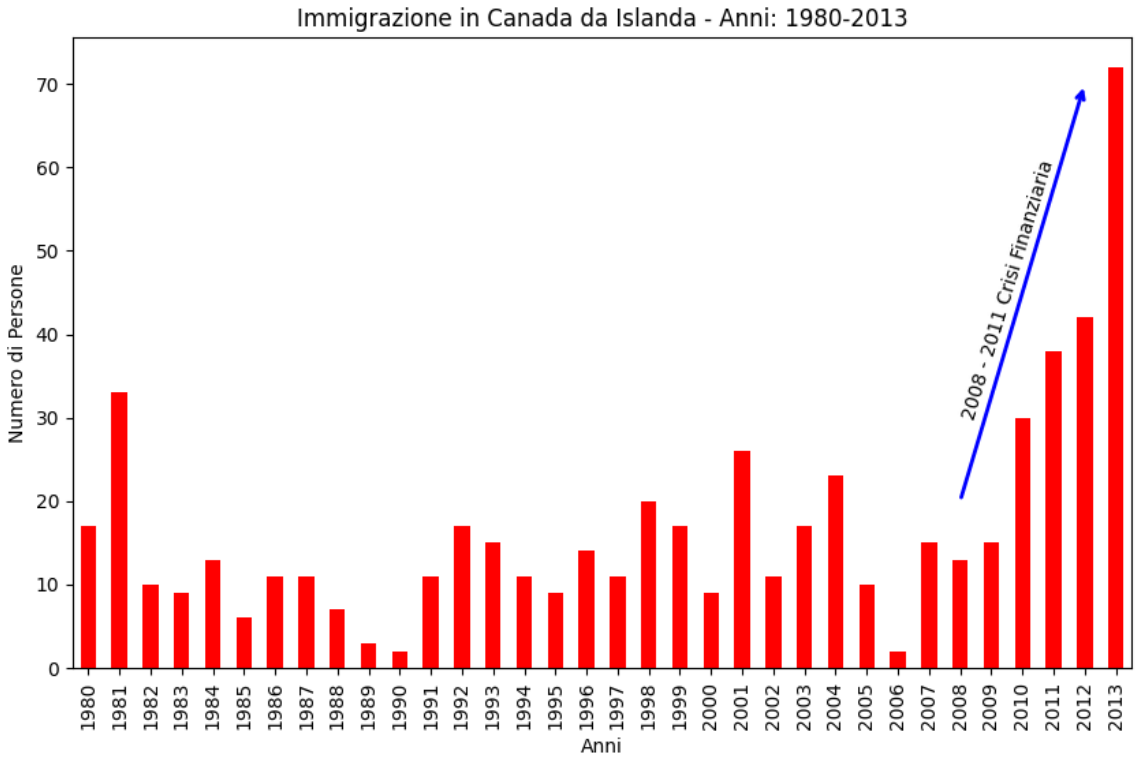
Bar chart

```
plt.annotate('',                # s: str. vuota per nessun testo
             xy=(32, 70),       # testa freccia al punto (anno = 2012 , persone = 70 )
             xytext=(28, 20), # testa freccia al punto (anno = 2008 , persone = 20 )
             xycoords='data', # usa il sistema di coordinate del sistema che si sta annotando
             arrowprops=dict(arrowstyle='->', connectionstyle='arc3', color='blue', lw=2)
            )
```



Bar chart

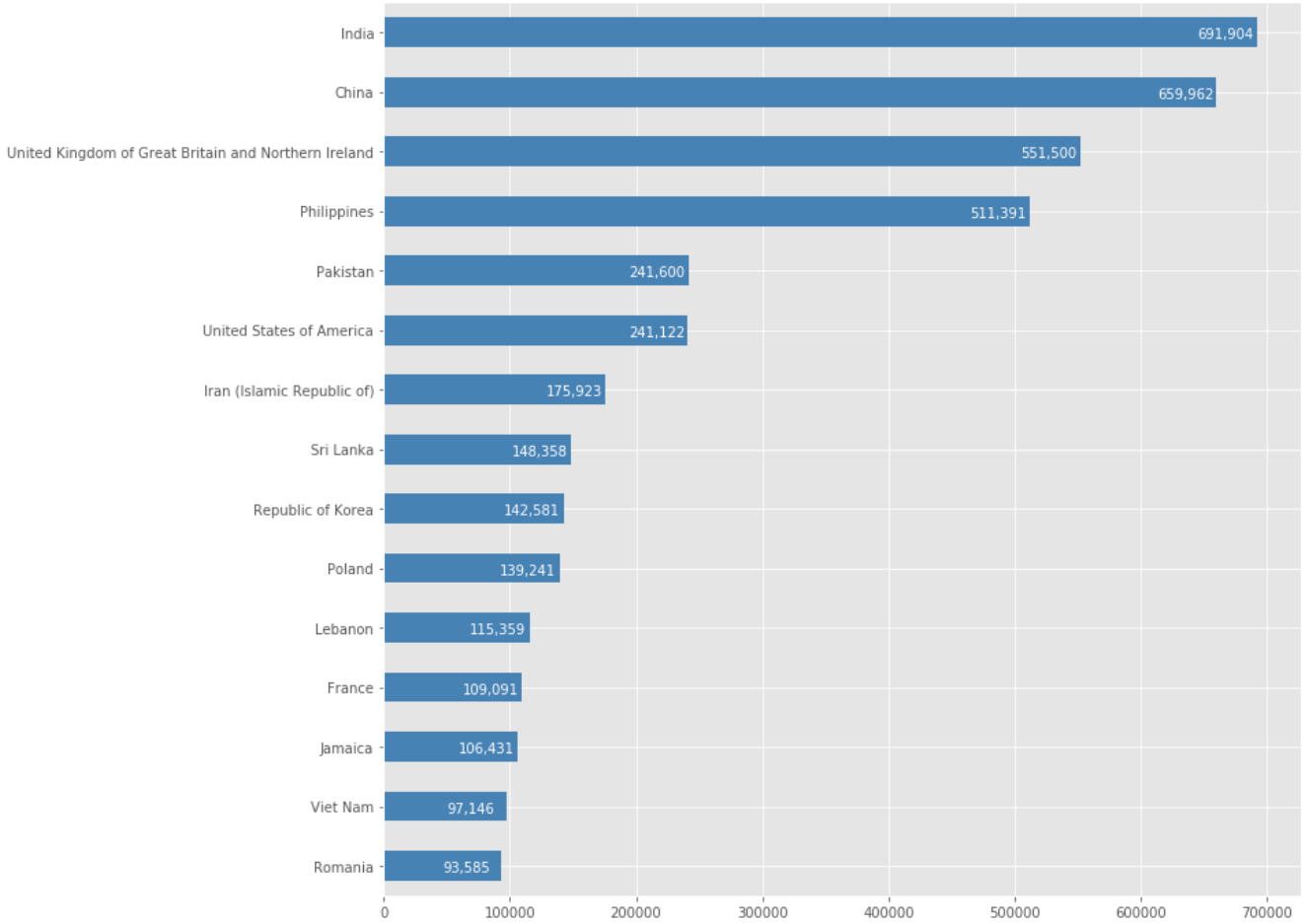
```
plt.annotate('2008 - 2011 Crisi Finanziaria', # testo da mostrare
             xy=(28,30),                      # inizio testo
             rotation=73,                     # ruota testo
            )
```



ICTAcademy
INFORMATION AND COMMUNICATIONS TECHNOLOGY

Bar chart

```
plt.annotate(label, xy=(value - offset_x, index - offset_y), color='white')
```

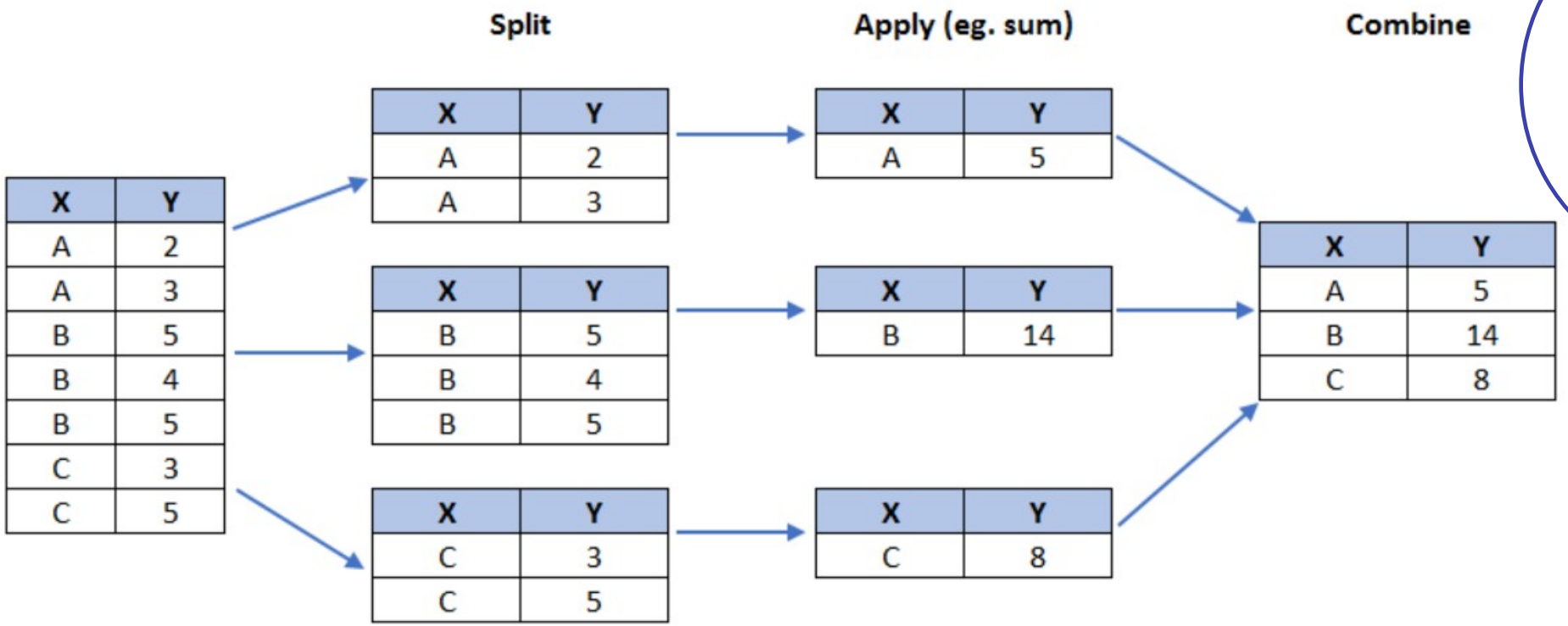


Pie chart

- Tipo di plot per rappresentare proporzioni numeriche dividendo un cerchio (o "pie") in fette proporzionali
- Uno use case (non l’unico): visualizzazione immediata valori di raggruppamenti (`df.groupby`)

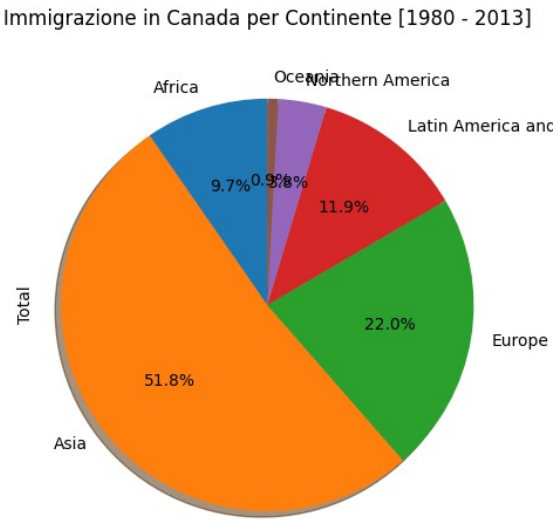
Pie chart

```
df_continents = df.groupby('Continent', axis=0).sum().loc[:, '1980':'Total']
```



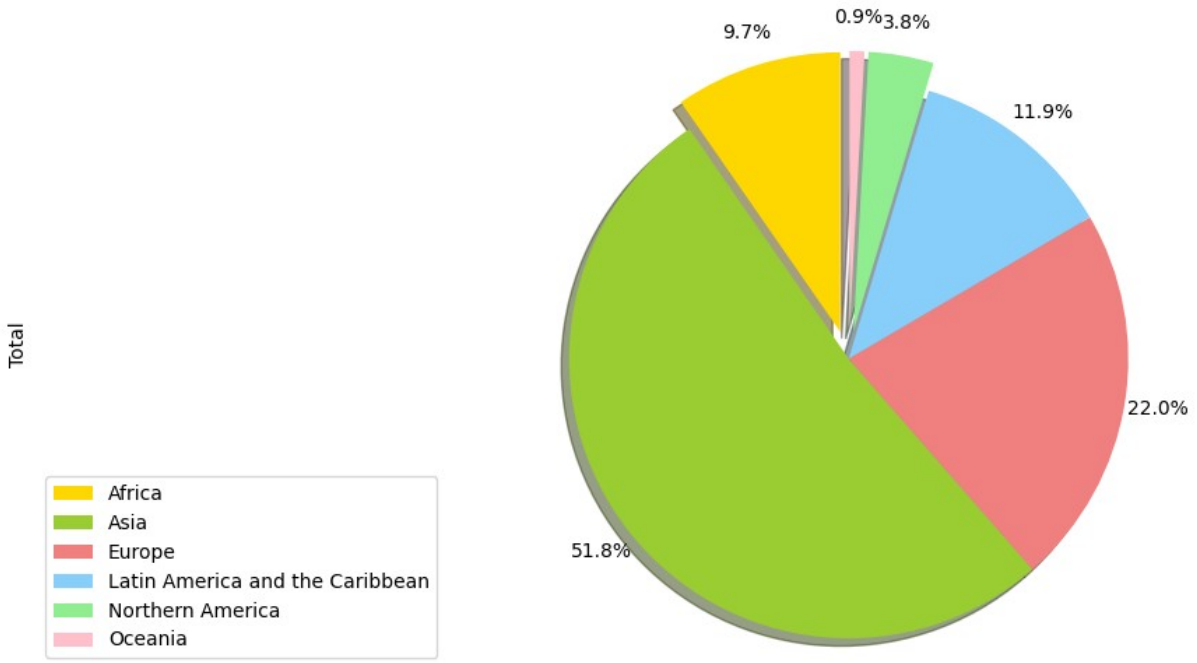
Pie chart

```
df_continents['Total'].plot(kind='pie',
                    figsize=(5, 6),
                    autopct='%1.1f%%', # aggiungi le percentuali
                    startangle=90,     # Angolo iniziale (Africa) a 90°
                    shadow=True,       # aggiunge un'ombra
                    )
plt.title('Immigrazione in Canada per Continente [1980 - 2013]')
plt.axis('equal') # Rende il pie chart della forma di un cerchio
```



API: pandas.DataFrame.plot

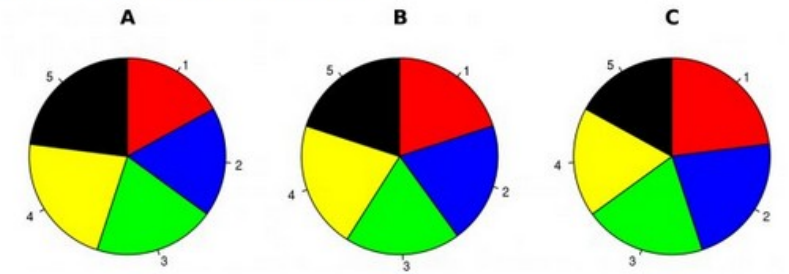
Pie chart


Total
Africa
Asia
Europe
Latin America and the Caribbean
Northern America
Oceania
9.7%
0.9%
3.8%
11.9%
22.0%
51.8%

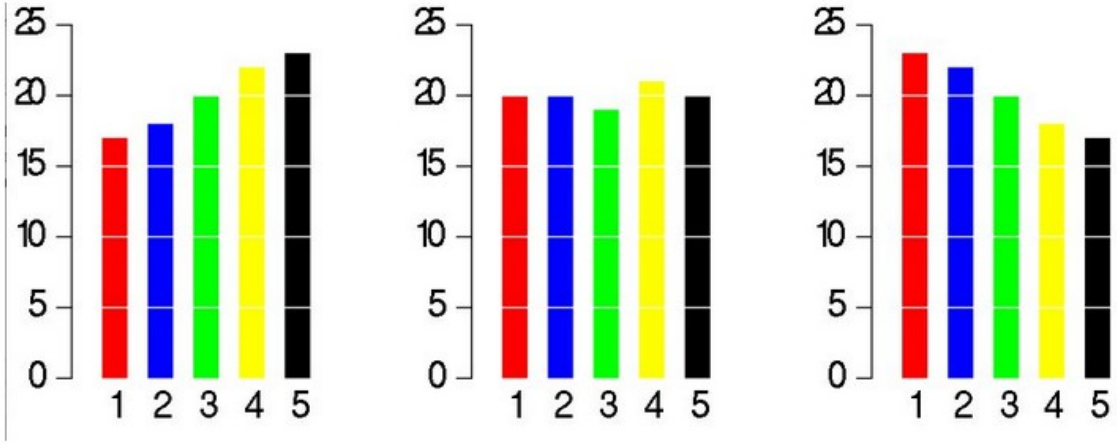
Problems With Pie Charts

There are some very vocal opponents to the use of pie charts under any circumstances. Most cite what they see as pie charts' failure to do their single job, accurately display data, with any consistency.

In an article on Business Insider called, *Pie Charts are the Worst*, Walter Hickey gives this example of how pie charts can fail to give meaningful insight into basic data:



Hickey contrasts the above graphs, which are meant to show how various political candidates' shares of the vote changed over time, with these bar graphs:



The bar graphs are using the same data set, but it's much simpler to see the patterns here than in the pie charts.

Hickey calls pie charts, "easily the worst way to convey information ever developed in the history of data visualization," which may be taking things a little too far, but this example makes it clear that they're not always the right choice.

If you don't have dramatic differences in your data percentages, it can be hard to see distribution on a pie chart.

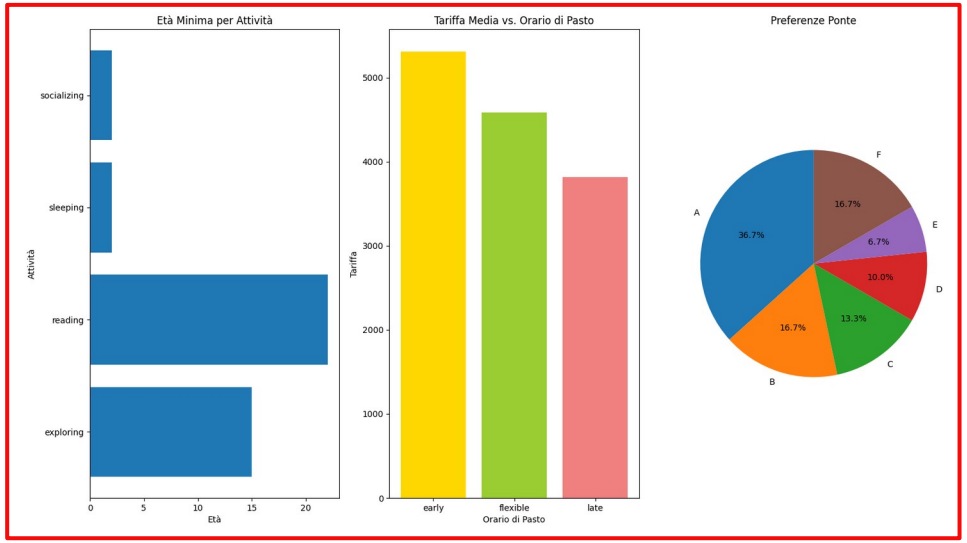
Additionally, the popular 3-D versions can drastically distort data, making pieces that are widely different appear close to the same size. As a rule, steer clear of these types of pie charts no matter how flashy they may be.

Una critica ai Pie chart...

Esercizio

/Lezione2/codice/titanic/titanic_data_pipeline_complete.py

```python
def run_pipeline(self) -> pd.DataFrame:
    """Esegue la pipeline completa"""
    # Carica dati
    db_df1, db_df2 = self.load_from_database()
    print("   -Letti dati da due tabelle su db")
    json_df = self.load_from_json()
    print("   -Letti dati da un file JSON")
    # Preprocessa dati
    merged_df = self.merge_data(db_df1, db_df2, json_df)
    expanded_df = self.expand_json_data(merged_df)
    print("   -Aggregazione ed espansione dati effettuate")
    clean_df = self.clean_data(expanded_df)
    print("   -Pulizia dati completata")
    # Visualizza risultati
    self.visualize(clean_df)
    print("   -Analisi e visualizzazione dati terminate")
    self.data = clean_df
    return clean_df
```



RIFERIMENTI BIBLIOGRAFICI

- click to pandas documentation
- click to matplotlib documentation