This is an analysis on the marketing data of a company's online store

```
# import the libraries we want to use
import pandas as pd
import matplotlib.pyplot as plt


# read the datatset into pandas
df = pd.read_csv('marketing_data.csv')


df
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumStorePurchases	NumWebVisitsMonth
0	1826	1970	Graduation	Divorced	84835.0	0	0	6/16/2014	0	189	...	6	1
1	1	1961	Graduation	Single	57091.0	0	0	6/15/2014	0	464	...	7	5
2	10476	1958	Graduation	Married	67267.0	0	1	5/13/2014	0	134	...	5	2
3	1386	1967	Graduation	Together	32474.0	1	1	5/11/2014	0	10	...	2	7
4	5371	1989	Graduation	Single	21474.0	1	0	4/8/2014	0	6	...	2	7
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2235	10142	1976	PhD	Divorced	66476.0	0	1	3/7/2013	99	372	...	11	4
2236	5263	1977	2n Cycle	Married	31056.0	1	0	1/22/2013	99	5	...	3	8
2237	22	1976	Graduation	Divorced	46310.0	1	0	12/3/2012	99	185	...	5	8
2238	528	1978	Graduation	Married	65819.0	0	0	11/29/2012	99	267	...	10	3
2239	4070	1969	PhD	Married	94871.0	0	2	9/1/2012	99	169	...	4	7

2240 rows × 28 columns

```
df.tail(20)
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	Num
2220	5871	1979	Master	Together	24401.0	0	0	8/31/2012	98	73	...	
2221	3846	1974	Graduation	Married	42557.0	0	1	8/29/2012	98	192	...	
2222	10001	1985	2n Cycle	Together	7500.0	1	0	8/1/2012	98	5	...	
2223	2831	1976	Graduation	Together	78416.0	0	1	6/27/2014	99	453	...	
2224	868	1966	Graduation	Married	44794.0	0	1	6/8/2014	99	54	...	
2225	7212	1966	Graduation	Married	44794.0	0	1	6/8/2014	99	54	...	
2226	1743	1974	Graduation	Single	69719.0	0	0	5/26/2014	99	273	...	
2227	2415	1962	Graduation	Together	62568.0	0	1	4/7/2014	99	362	...	
2228	7947	1969	Graduation	Married	42231.0	1	1	3/25/2014	99	24	...	
2229	2106	1974	2n Cycle	Married	20130.0	0	0	3/17/2014	99	0	...	
2230	3363	1974	2n Cycle	Married	20130.0	0	0	3/17/2014	99	0	...	
2231	8595	1973	Graduation	Widow	42429.0	0	1	2/11/2014	99	55	...	
2232	7232	1973	Graduation	Widow	42429.0	0	1	2/11/2014	99	55	...	
2233	7829	1900	2n Cycle	Divorced	36640.0	1	0	9/26/2013	99	15	...	
2234	9977	1973	Graduation	Divorced	78901.0	0	1	9/17/2013	99	321	...	
2235	10142	1976	PhD	Divorced	66476.0	0	1	3/7/2013	99	372	...	
2236	5263	1977	2n Cycle	Married	31056.0	1	0	1/22/2013	99	5	...	
2237	22	1976	Graduation	Divorced	46310.0	1	0	12/3/2012	99	185	...	
2238	528	1978	Graduation	Married	65819.0	0	0	11/29/2012	99	267	...	
2239	4070	1969	PhD	Married	94871.0	0	2	9/1/2012	99	169	...	

20 rows × 28 columns

```
df.loc[100:120]
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumS
100	6715	1948	PhD	Single	60200.0	0	1	1/2/2013	3	502	...	
101	10676	1982	Graduation	Married	63211.0	0	0	11/2/2012	3	145	...	
102	1041	1973	PhD	Single	48432.0	0	1	10/18/2012	3	322	...	
103	492	1973	PhD	YOLO	48432.0	0	1	10/18/2012	3	322	...	
104	11133	1973	PhD	YOLO	48432.0	0	1	10/18/2012	3	322	...	
105	4491	1949	Master	Married	62845.0	1	1	10/1/2012	3	1099	...	
106	873	1949	Master	Married	62845.0	1	1	10/1/2012	3	1099	...	
107	1631	1965	PhD	Together	65220.0	0	0	9/3/2012	3	890	...	
108	8278	1990	PhD	Married	74214.0	0	0	8/26/2012	3	863	...	
109	6815	1980	2n Cycle	Married	96547.0	0	0	5/23/2014	4	448	...	
110	254	1955	Graduation	Together	53863.0	0	1	5/17/2014	4	399	...	
111	1349	1970	Graduation	Married	50447.0	2	0	4/21/2014	4	85	...	
112	2534	1953	Graduation	Married	37716.0	0	1	4/21/2014	4	97	...	
113	2130	1982	Graduation	Together	45203.0	2	0	3/23/2014	4	35	...	
114	2296	1975	Master	Married	37368.0	1	0	12/16/2013	4	3	...	
115	3799	1955	Graduation	Married	67225.0	0	1	11/26/2013	4	315	...	
116	11084	1976	Master	Together	65104.0	0	1	11/14/2013	4	738	...	
117	5172	1976	Master	Together	65104.0	0	1	11/14/2013	4	738	...	
118	9504	1949	Master	Married	81698.0	0	0	11/6/2013	4	179	...	
119	850	1968	Graduation	Single	70566.0	0	1	10/6/2013	4	381	...	
120	4477	1958	Graduation	Together	69096.0	0	1	9/27/2013	4	247	...	

21 rows × 28 columns

```
# listing out the columns in the dataset
df.columns.tolist()
```

```
['ID',
 'Year_Birth',
 'Education',
 'Marital_Status',
```

```
        'Income',
        'Kidhome',
        'Teenhome',
        'Dt_Customer',
        'Recency',
        'MntWines',
        'MntFruits',
        'MntMeatProducts',
        'MntFishProducts',
        'MntSweetProducts',
        'MntGoldProds',
        'NumDealsPurchases',
        'NumWebPurchases',
        'NumCatalogPurchases',
        'NumStorePurchases',
        'NumWebVisitsMonth',
        'AcceptedCmp3',
        'AcceptedCmp4',
        'AcceptedCmp5',
        'AcceptedCmp1',
        'AcceptedCmp2',
        'Response',
        'Complain',
        'Country']
```

```
# check for quality of the dataset
df.shape
```

```
    (2240, 28)
```

```
df.isna().sum()
```

```
    ID                     0
    Year_Birth             0
    Education              0
    Marital_Status         0
    Income                24
    Kidhome                0
    Teenhome               0
    Dt_Customer            0
    Recency                0
    MntWines               0
    MntFruits              0
    MntMeatProducts        0
    MntFishProducts        0
    MntSweetProducts       0
    MntGoldProds           0
    NumDealsPurchases      0
    NumWebPurchases        0
    NumCatalogPurchases    0
    NumStorePurchases      0
    NumWebVisitsMonth      0
```

```
AcceptedCmp3            0
AcceptedCmp4            0
AcceptedCmp5            0
AcceptedCmp1            0
AcceptedCmp2            0
Response                0
Complain                0
Country                 0
dtype: int64
```

```
# filling the null values with zero
df['Income'].fillna(0, inplace = True)
```

```
df[df['tot_amt_goods'] > 1200]
```

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts
6	4073	1954	2n Cycle	Married	63564.0	1/29/2014	0	769	80	252
10	2079	1947	2n Cycle	Married	81044.0	12/27/2013	0	450	26	535
31	9264	1986	Graduation	Married	79529.0	4/27/2014	1	423	42	706
35	7962	1987	PhD	Single	95169.0	10/9/2013	1	1285	21	449
38	3725	1961	PhD	Single	84865.0	5/9/2013	1	1248	16	349
...	...	...	...	...	...	...	...	...	...	...
2213	5602	1989	PhD	Together	66973.0	5/17/2013	98	466	22	432
2215	9645	1968	Graduation	Married	64590.0	10/14/2012	98	920	138	168
2218	4974	1970	Graduation	Single	83273.0	9/25/2012	98	433	89	650
2219	5687	1980	Graduation	Divorced	81702.0	9/23/2012	98	563	50	774
2238	528	1978	Graduation	Married	65819.0	11/29/2012	99	267	38	701

417 rows × 30 columns

```
df.isna().sum()
```

```
ID                    0
Year_Birth            0
Education             0
Marital_Status        0
Income                0
Kidhome               0
Teenhome              0
Dt_Customer           0
```

```
Recency               0
MntWines              0
MntFruits             0
MntMeatProducts       0
MntFishProducts       0
MntSweetProducts      0
MntGoldProds          0
NumDealsPurchases     0
NumWebPurchases       0
NumCatalogPurchases   0
NumStorePurchases     0
NumWebVisitsMonth     0
AcceptedCmp3          0
AcceptedCmp4          0
AcceptedCmp5          0
AcceptedCmp1          0
AcceptedCmp2          0
Response              0
Complain              0
Country               0
dtype: int64
```

```
df.dtypes
```

```
ID                     int64
Year_Birth             int64
Education             object
Marital_Status        object
Income               float64
Kidhome                int64
Teenhome               int64
Dt_Customer           object
Recency                int64
MntWines               int64
MntFruits              int64
MntMeatProducts        int64
MntFishProducts        int64
MntSweetProducts       int64
MntGoldProds           int64
NumDealsPurchases      int64
NumWebPurchases        int64
NumCatalogPurchases    int64
NumStorePurchases      int64
NumWebVisitsMonth      int64
AcceptedCmp3           int64
AcceptedCmp4           int64
AcceptedCmp5           int64
AcceptedCmp1           int64
AcceptedCmp2           int64
Response               int64
Complain               int64
Country               object
dtype: object
```

```
df.nunique()
```

```
ID                     2240
Year_Birth               59
Education                 5
Marital_Status            8
Income                 1975
Kidhome                   3
Teenhome                  3
Dt_Customer             663
Recency                 100
MntWines                776
MntFruits               158
MntMeatProducts         558
MntFishProducts         182
MntSweetProducts        177
MntGoldProds            213
NumDealsPurchases        15
NumWebPurchases          15
NumCatalogPurchases      14
NumStorePurchases        14
NumWebVisitsMonth        16
AcceptedCmp3              2
AcceptedCmp4              2
AcceptedCmp5              2
AcceptedCmp1              2
AcceptedCmp2              2
Response                  2
Complain                  2
Country                   8
dtype: int64
```

```
# creating a new column for the Age of the customers
def cust_age(year):
    age = 2023 - year
    return age
Ages = cust_age(df['Year_Birth'])
print(Ages)
```

```
0       53
1       62
2       65
3       56
4       34
       ..
2235    47
2236    46
2237    47
2238    45
```

```
2239    54
Name: Year_Birth, Length: 2240, dtype: int64
```

```
df['cust_age'] = Ages
```

```
df.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWeb
0	1826	1970	Graduation	Divorced	84835.0	0	0	6/16/2014	0	189	...	
1	1	1961	Graduation	Single	57091.0	0	0	6/15/2014	0	464	...	
2	10476	1958	Graduation	Married	67267.0	0	1	5/13/2014	0	134	...	
3	1386	1967	Graduation	Together	32474.0	1	1	5/11/2014	0	10	...	
4	5371	1989	Graduation	Single	21474.0	1	0	4/8/2014	0	6	...	

5 rows × 29 columns

```
# checking for the unique values in the age column
df['cust_age'].unique()
```

```
array([ 53,  62,  65,  56,  34,  69,  76,  44,  64,  42,  54,  46,  63,
        57,  47,  58,  67,  48,  52,  37,  51,  49,  33,  36,  39,  55,
        68,  40,  50,  45,  71,  61,  59,  41,  60,  66,  43,  78,  74,
        75,  70,  77,  38,  31,  79,  72,  35,  73,  29,  30,  32, 130,
        27,  28, 124,  80,  82,  83, 123])
```

```
# checking for the maximum and minimum age
a = [df['cust_age'].min(), df['cust_age'].max()]
```

```
a
```

```
[27, 130]
```

```
# grouping the ages into bands
def age_band(age):
    if age >= 25 and age <= 35:
        ageband = '25-35'
    elif age > 35 and age <= 50:
        ageband = '35-50'
    else:
        ageband = 'above 50'
    return ageband

df['age_band'] = df['cust_age'].apply(age_band)


# combining Kidhome and Teenhome into one column
df['kids'] = df[['Kidhome','Teenhome']].sum(axis = 1)
df.drop(['Kidhome','Teenhome'], axis = 1, inplace = True)


df['tot_amt_goods'] = df[['MntWines','MntFruits','MntMeatProducts','MntFishProducts','MntSweetProducts','MntGoldProds']].sum(axis = 1)


df.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	..
0	1826	1970	Graduation	Divorced	84835.0	6/16/2014	0	189	104	379	.
1	1	1961	Graduation	Single	57091.0	6/15/2014	0	464	5	64	.
2	10476	1958	Graduation	Married	67267.0	5/13/2014	0	134	11	59	.
3	1386	1967	Graduation	Together	32474.0	5/11/2014	0	10	0	1	.
4	5371	1989	Graduation	Single	21474.0	4/8/2014	0	6	16	24	.

5 rows × 30 columns

```
# the distribution of marital status according to their volume of purchase
vol_goods = df.groupby('Marital_Status').sum()['tot_amt_goods']
```

```
<ipython-input-25-71700d242977>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_o
  vol_goods = df.groupby('Marital_Status').sum()['tot_amt_goods']
```

```
vol_goods
```

```
Marital_Status
Absurd        2385
Alone          770
Divorced    141666
```

```
Married     510453
Single      291112
Together    352865
Widow        56889
YOLO           848
Name: tot_amt_goods, dtype: int64
```

```
vol_goods.plot(kind = 'line', marker = 'o')
plt.title('the distribution of goods purchased according to their marital status')
plt.xlabel('marriage_status')
plt.ylabel('volume_goods')
```

```
Text(0, 0.5, 'volume_goods')
```


```
# the average distribution of income according to their marital status
av_distro = df.groupby('Marital_Status').mean()['Income'].sort_values(ascending = False)
```

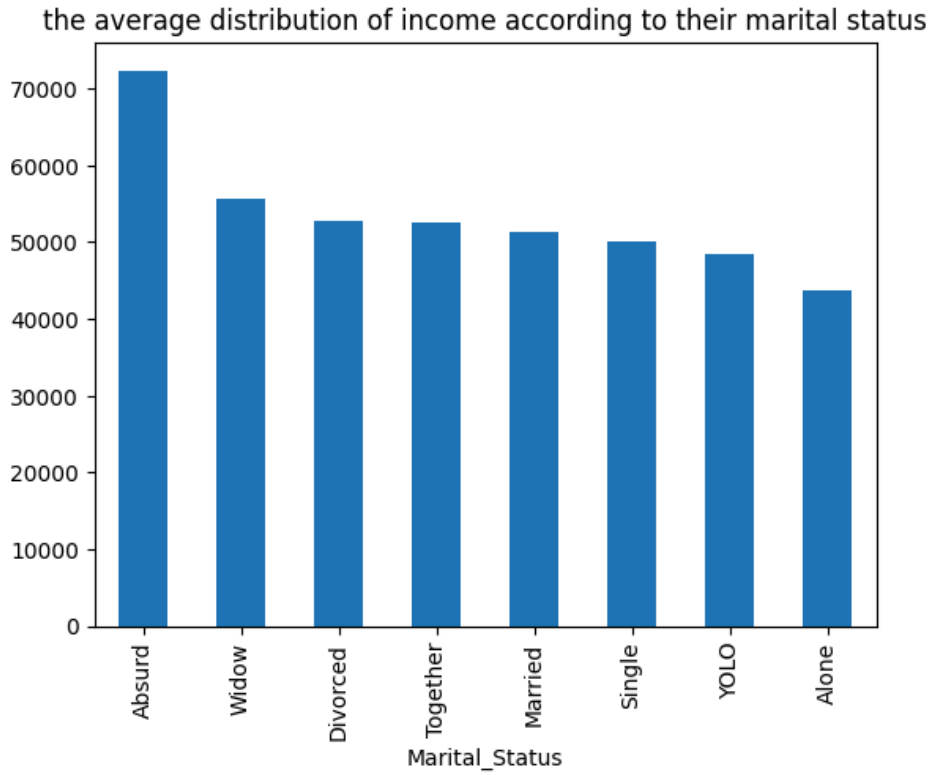
```
<ipython-input-37-19902e1ceac0>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_
  av_distro = df.groupby('Marital_Status').mean()['Income'].sort_values(ascending = False)
```

```
av_distro
```

```
Marital_Status
Absurd      72365.500000
Widow       55748.025974
Divorced    52834.228448
Together    52602.915517
Married     51305.910880
Single      50039.187500
YOLO        48432.000000
Alone       43789.000000
Name: Income, dtype: float64
```

```
av_distro.plot(kind = 'bar')
plt.title('the average distribution of income according to their marital status')
```

```
Text(0.5, 1.0, 'the average distribution of income according to their marital status')
```




```
# the distribution of goods purchased according to their age band
top_spenders = df.groupby('age_band').sum()['tot_amt_goods']
```

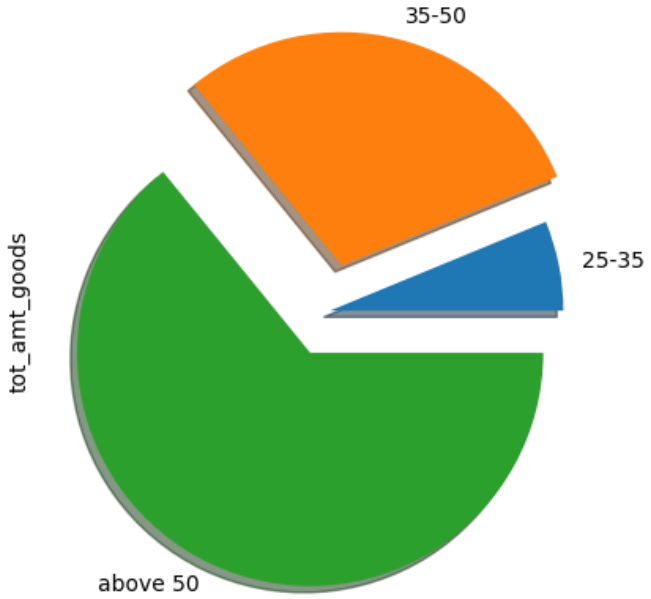
```
<ipython-input-40-bca15c184e1b>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_o
  top_spenders = df.groupby('age_band').sum()['tot_amt_goods']
```

```
top_spenders
```

```
age_band
25-35        84312
35-50       402439
above 50    870237
Name: tot_amt_goods, dtype: int64
```

```
P_explode = [0, 0.2, 0.2]
top_spenders.plot(kind = 'pie', explode = P_explode, shadow = True)
```

```
<Axes: ylabel='tot_amt_goods'>
```




```
df.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	..
0	1826	1970	Graduation	Divorced	84835.0	6/16/2014	0	189	104	379	.
1	1	1961	Graduation	Single	57091.0	6/15/2014	0	464	5	64	.
2	10476	1958	Graduation	Married	67267.0	5/13/2014	0	134	11	59	.
3	1386	1967	Graduation	Together	32474.0	5/11/2014	0	10	0	1	.
4	5371	1989	Graduation	Single	21474.0	4/8/2014	0	6	16	24	.

5 rows × 30 columns

```
df['Income'] = df[' Income ']
```

```
df['Income']
```

```
0       84835.0
1       57091.0
2       67267.0
3       32474.0
4       21474.0
         ...
2235    66476.0
2236    31056.0
2237    46310.0
2238    65819.0
2239    94871.0
Name: Income, Length: 2240, dtype: float64
```

```
df.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	..
0	1826	1970	Graduation	Divorced	84835.0	6/16/2014	0	189	104	379	.
1	1	1961	Graduation	Single	57091.0	6/15/2014	0	464	5	64	.
2	10476	1958	Graduation	Married	67267.0	5/13/2014	0	134	11	59	.
3	1386	1967	Graduation	Together	32474.0	5/11/2014	0	10	0	1	.
4	5371	1989	Graduation	Single	21474.0	4/8/2014	0	6	16	24	.

5 rows × 30 columns

```
df['Education'].unique()
```

```
array(['Graduation', 'PhD', '2n Cycle', 'Master', 'Basic'], dtype=object)
```

```
# the distribution of age band according to their value of income
df.groupby('age_band').mean()['Income']
```

```
<ipython-input-38-792bbd47793a>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_
  df.groupby('age_band').mean()['Income']
age_band
25-35       48676.975000
35-50       47757.717853
above 50    54347.793783
Name: Income, dtype: float64
```

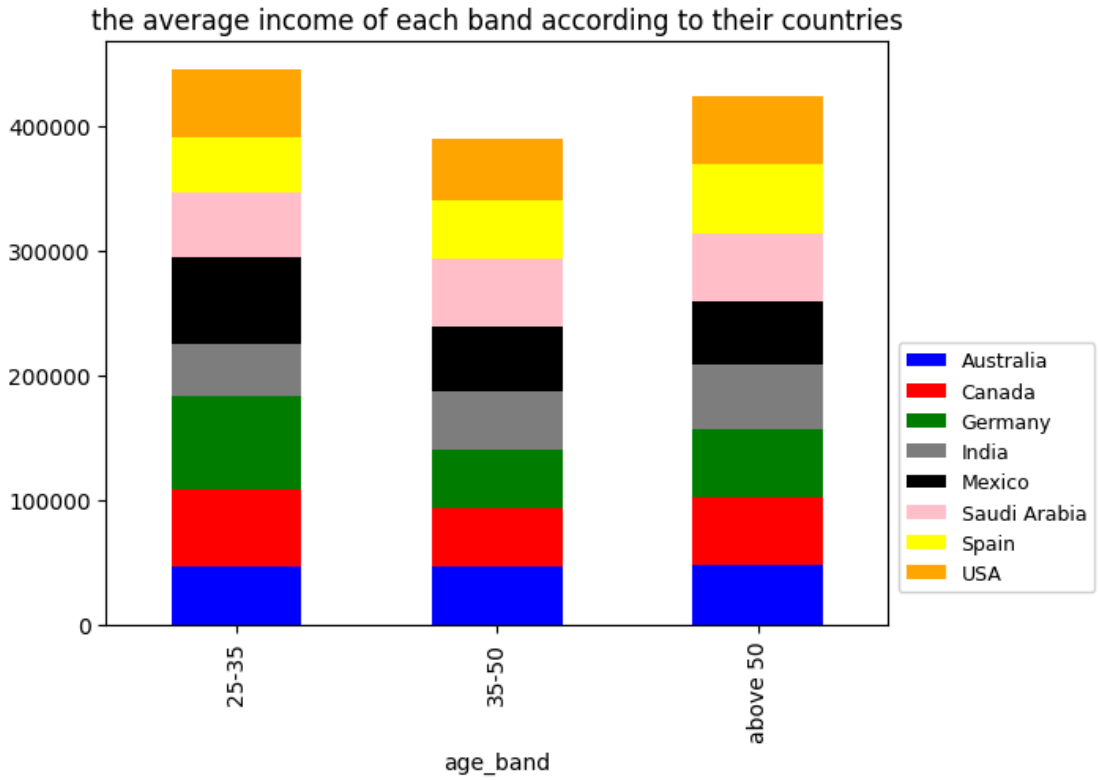
```
# the average income of each band according to their countries
average_income = df.pivot_table(index = 'age_band', columns = 'Country', values = 'Income')
```

```
average_income
```

Country	Australia	Canada	Germany	India	Mexico	Saudi Arabia	Spain	USA
age_band								
25-35	46129.800000	62870.416667	74724.000000	40853.800000	70515.0	51326.708333	44920.470588	53987.800000
35-50	46732.392157	47366.080460	46496.266667	46207.063492	52614.0	54427.073171	46093.439589	49258.619048
above 50	48187.676768	54651.893491	53432.150685	52593.728571	49912.0	55534.810526	55173.726718	54122.016129

```
average_income.plot(kind = 'bar', stacked=True, color = ['blue', 'red','green','grey','black','pink','yellow','Orange'])
plt.title('the average income of each band according to their countries')
plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.5))
```

```
<matplotlib.legend.Legend at 0x7e5a3513b9a0>
```




```
# total amount of goods purchased by country according to the amount of kids they have
df.pivot_table(index = 'kids', columns = 'Country', values = 'tot_amt_goods', aggfunc = 'sum')
```

Country	Australia	Canada	Germany	India	Mexico	Saudi Arabia	Spain	USA
kids								
0	40418.0	86012.0	46033.0	42133.0	1258.0	112532.0	346267.0	30994.0
1	38654.0	66957.0	22951.0	29723.0	1864.0	79361.0	263384.0	30349.0
2	9740.0	13050.0	3764.0	7462.0	NaN	16527.0	48285.0	4716.0
3	951.0	2513.0	2165.0	167.0	NaN	2651.0	4284.0	1823.0

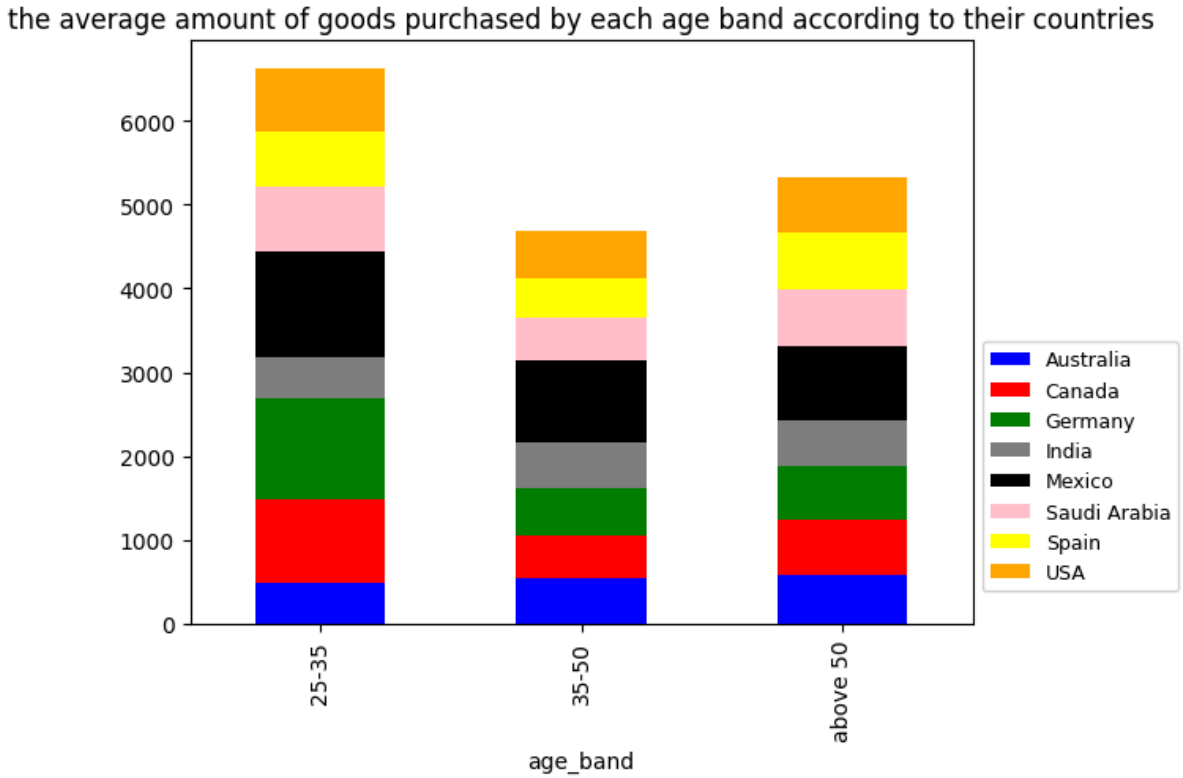
```
# pivot tables gives you the average values by default except you specify otherwise
# the average amount of goods purchased by each age band according to their countries
av_purchase = df.pivot_table(index = 'age_band', columns = 'Country', values = 'tot_amt_goods')
```

```
av_purchase
```

Country	Australia	Canada	Germany	India	Mexico	Saudi Arabia	Spain	USA
age_band								
25-35	491.800000	999.333333	1193.500000	498.400000	1258.0	762.625000	671.254902	748.800000
35-50	547.921569	509.908046	557.222222	539.476190	990.0	500.276423	475.609254	560.357143
above 50	574.757576	663.775148	650.013699	543.171429	874.0	690.705263	676.296183	654.887097

```
av_purchase.plot(kind = 'bar', stacked=True, color = ['blue', 'red','green','grey','black','pink','yellow','orange'])
plt.title('the average amount of goods purchased by each age band according to their countries')
plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.5))
```

```
<matplotlib.legend.Legend at 0x7de5bb793e80>
```



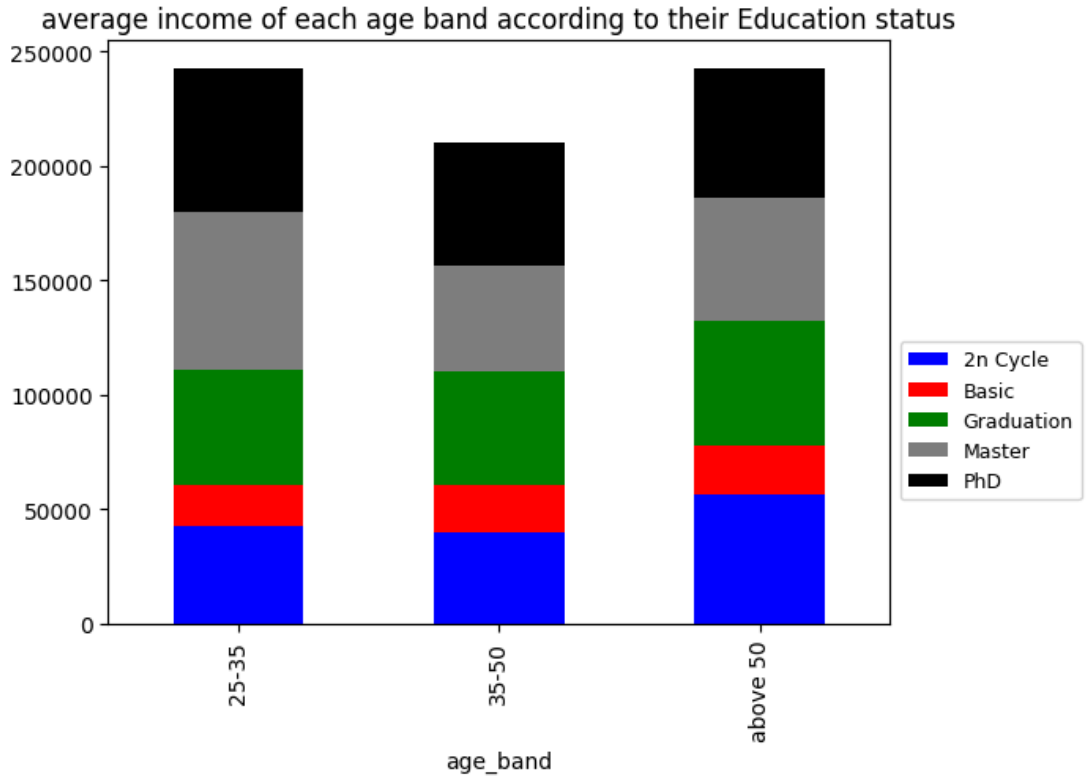

```
# average income of each age band according to their Education status
av_income = df.pivot_table(index = 'age_band', columns = 'Education', values = 'Income')
```

```
av_income
```

Education	2n Cycle	Basic	Graduation	Master	PhD
age_band					
25-35	42732.944444	17923.818182	50146.323944	68688.700000	62760.600000
35-50	40024.336634	20403.862069	49950.279126	46051.441441	53571.195946
above 50	56130.797619	21976.000000	53875.827640	54282.341365	56249.246951

```
av_income.plot(kind = 'bar', stacked=True, color = ['blue', 'red','green','grey','black'])
plt.title('average income of each age band according to their Education status')
plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.5))
```

```
<matplotlib.legend.Legend at 0x7de5bb6a00a0>
```




```
# total goods purchased by each age band according to their Education status
total_goods = df.pivot_table(index = 'age_band', columns = 'Education', values = 'tot_amt_goods', aggfunc = 'sum')
```

```
total_goods
```

Education	2n Cycle	Basic	Graduation	Master	PhD
age_band					
25-35	8448	781	47697	13637	13749
35-50	36597	2433	219217	53086	91106
above 50	55750	1203	431712	159636	221936

```
total_goods.plot(kind = 'bar', stacked=True, color = ['blue', 'red','green','grey','black'])
plt.title('total goods purchased by each age band according to their Education status')
plt.legend(loc='upper left', fontsize=9, bbox_to_anchor=(1, 0.5))
```

```
<matplotlib.legend.Legend at 0x7de5bb76fa00>
```