پروژه فرار مالیاتی

فهرست مطالب

- فراخوانی کتابخانه های مورد استفاده

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

- فراخوانی شناسنامه داده مورد استفاده

In [2]:
```
df_describe = pd.read_excel(r"E:\data science bootcamp\project\شناسه داده دو درصد سایت.x
```

In [3]:
```
df_describe
```

Out[3]:

	متغیر	توضیحات
0	id	شناسه
1	Parent_Id	شناسه سرپرست
2	Age	سن
3	GenderId	جنسیت
4	Dashboard_postalcode7Digits	رقم ابتدای کد پستی 7
5	SabteAhval_countyname	شهرستان
6	SabteAhval_provincename	استان
7	isurban	شهری/روستایی
8	Decile	دهک رفاهی
9	Percentile	صدک رفاهی
10	Has_SoeTaghzie	آیا سو تغذیه دارد
11	Has_Saham_Edalat	دارای سهام عدالت
12	HasMojavezSenfi	دارای مجوز صنفی
13	ISBimarKhas	آیا بیماری خاص دارد؟
14	IsMalool	آیا معلولیت دارد؟
15	Malool_shedat	شدت معلولیت
16	IsBehzisti_AfzayeshMostamari	آیا تحت پوشش بهزیستی است؟
17	IsKomite_AfzayeshMostamari	آیا تحت پوشش کمیته امداداست؟
18	IsKomite_AfzayeshMostamariSayer	آیا تحت پوشش کمیته امداداست؟
19	is_bime_darman	دارای بیمه درمان
20	IsBimePardaz	بیمه پرداز
21	ISKarmanddolat_1402	کارمند دولت
22	IsRetired_Asli	بازنشسته اصلی
23	IsRetired_Tabaie	بازنشسته تبعی
24	TripCountAirNonPilgrimage_95to99	تعداد سفر های هوایی غیر زیارتی از 95 الی 99
25	TripCountAirPilgrimage_95to99	تعداد سفر های هوایی زیارتی از 95 تا99
26	TripCountNonAirNonPilgrimage_95to99	تعداد سفر های غیر زیارتی غیر هوایی 95 الی 99
27	TripCountNonAirPilgrimage_95to99	تعداد سفر های غیر هوایی زیارتی 95 الی 99
28	CardPerMonth_1398	خرید کارت ماهانه سال 1398 (ریال)
29	CardPerMonth_1399	خرید کارت ماهانه سال 1399 (ریال)

	متغیر	توضیحات
30	CardPerMonth_1400	خرید کارت ماهانه سال 1400 (ریال)
31	CardPerMonth_1401	خرید کارت ماهانه سال 1401 (ریال)
32	CardPerMonth_1402	خرید کارت ماهانه سال 1402 (ریال)
33	CardBeCardPerMonth_1401	گردش حساب-کارت به کارت ماهانه 1401 (ریال)
34	CardBeCardPerMonth_1402	گردش حساب-کارت به کارت ماهانه 1402 (ریال)
35	PayaPerMonth_1401	گردش حساب-پایا ماهیانه 1401 (ریال)
36	PayaPerMonth_1402	گردش حساب-پایا ماهیانه 1402 (ریال)
37	SatnaPerMonth_1401	گردش حساب-ساتنا ماهیانه 1401 (ریال)
38	SatnaPerMonth_1402	گردش حساب-ساتنا ماهیانه 1402 (ریال)
39	CarsCount	تعداد خودرو
40	CarsPrice	ارزش خودرو (ریال)
41	Daramad	مجموع درآمد های ثبتی (ریال)
42	Bourse_NetPortfoValue	ازرش پورتفوی بورسی (ریال)
43	Variz_1400	واریز سال 1400 (ریال)
44	MandehAkhar_1399	مانده انتهای سال 1399 (ریال)
45	MandehAkhar_1400	مانده انتهای سال 1400 (ریال)
46	MandehAval_1399	مانده ابتدای سال 1399 (ریال)
47	MandehAval_1400	مانده ابتدای سال 1400 (ریال)

- فراخوانی داده مورد استفاده

In [4]:
```
df = pd.read_csv(r"E:\data science bootcamp\project\nemone_2_darsadi_1402.csv")
df
```

Out[4]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_she
0	717641288407	327754746508	2	9.0	NaN	NaN	N
1	84729392521	84729392521	2	30.0	NaN	NaN	N
2	480898026296	480898026296	1	33.0	NaN	NaN	N
3	72903495587	824385263640	2	54.0	NaN	NaN	N
4	597960900957	493995376717	2	35.0	NaN	NaN	N
...	...	...	...	...	...	...	
1697811	53326817442	557444773010	1	12.0	NaN	NaN	N
1697812	208062992297	408375687751	2	33.0	NaN	NaN	N
1697813	360855746695	679652402477	1	41.0	NaN	NaN	N
1697814	513741239072	218828740607	2	54.0	NaN	NaN	N
1697815	231940330996	231940330996	1	27.0	NaN	NaN	N

1697816 rows × 48 columns

ایجاد ستون 'Yaraneh'

In [5]:
```
df['Yaraneh'] = (df['Decile'].apply(lambda x: True if x < 4 else False))
df.head()
```

Out[5]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_shedat	Da
0	717641288407	327754746508	2	9.0	NaN	NaN	NaN	
1	84729392521	84729392521	2	30.0	NaN	NaN	NaN	
2	480898026296	480898026296	1	33.0	NaN	NaN	NaN	
3	72903495587	824385263640	2	54.0	NaN	NaN	NaN	
4	597960900957	493995376717	2	35.0	NaN	NaN	NaN	

5 rows × 49 columns

همسان سازی سال ها مطابق با نرخ تورم اعلامی از بانک مرکزی و تبدیل به سال 1402

سال 1398 به سال 1402

In [6]:
```
def inflation_1398(x):
    x = x * 147.1/100
    x = x * 146.2/100
    x = x * 146.5/100
    x = x * 150/100
    return x
```

In [7]:
```
df[ "CardPerMonth_1398"] =df["CardPerMonth_1398"].apply(inflation_1398)
```

سال 1399 به سال 1402

In [8]:
```
def inflation_1399(x):
    x = x * 146.2/100
    x = x * 146.5/100
    x = x * 150/100
    return x
```

In [9]:
```
df[ "CardPerMonth_1399"] =df["CardPerMonth_1399"].apply(inflation_1399)
df[ "MandehAkhar_1399"] =df["MandehAkhar_1399"].apply(inflation_1399)
df[ "MandehAval_1399"] =df["MandehAval_1399"].apply(inflation_1399)
```

سال 1400 به 1402

In [10]:
```
def inflation_1400(x):
    x = x * 146.5/100
    x = x * 150/100
    return x
```

In [11]:
```
df[ "CardPerMonth_1400"] =df["CardPerMonth_1400"].apply(inflation_1400)
df[ "MandehAkhar_1400"] =df["MandehAkhar_1400"].apply(inflation_1400)
df[ "MandehAval_1400"] =df["MandehAval_1400"].apply(inflation_1400)
df[ "Variz_1400"] =df["Variz_1400"].apply(inflation_1400)
```

سال 1401 به سال 1402

In [12]:
```
def inflation_1401(x):
    x = x * 150/100
    return x
```

In [13]:
```
df[ "CardPerMonth_1401"] = df["CardPerMonth_1401"].apply(inflation_1401)
df[ "CardBeCardPerMonth_1401"] = df["CardBeCardPerMonth_1401"].apply(inflation_1401
df[ "PayaPerMonth_1401"] = df["PayaPerMonth_1401"].apply(inflation_1401)
df[ "SatnaPerMonth_1401"] = df["SatnaPerMonth_1401"].apply(inflation_1401)
```

In [14]:
```
df
```

Out[14]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_she
0	717641288407	327754746508	2	9.0	NaN	NaN	N
1	84729392521	84729392521	2	30.0	NaN	NaN	N
2	480898026296	480898026296	1	33.0	NaN	NaN	N
3	72903495587	824385263640	2	54.0	NaN	NaN	N
4	597960900957	493995376717	2	35.0	NaN	NaN	N
...	...	...	...	...	...	...	
1697811	53326817442	557444773010	1	12.0	NaN	NaN	N
1697812	208062992297	408375687751	2	33.0	NaN	NaN	N
1697813	360855746695	679652402477	1	41.0	NaN	NaN	N
1697814	513741239072	218828740607	2	54.0	NaN	NaN	N
1697815	231940330996	231940330996	1	27.0	NaN	NaN	N

1697816 rows × 49 columns

معیار های ستون ثروت

1- هزینه سفر ها

2- دارایی های ثابت

3- ثروت های نقدی جاری

1- هزینه سفر ها(total_trip_cost)

برای محاسبه هزینه هر نوع سفر، می‌توان مراحل زیر را دنبال کرد:

1. **سفرهای هوایی زیارتی:** هزینه بلیط هواپیما، اقامت و غذا در مقصدهای زیارتی مانند مکه یا مشهد به‌طور میانگین محاسبه می‌شود.

2. **سفرهای هوایی غیرزیارتی:** هزینه بلیط، اقامت، غذا و تفریحات در مقصدهای تفریحی یا کاری در نظر گرفته می‌شود.

3. **سفرهای غیرهوایی زیارتی:** هزینه حمل و نقل (اتوبوس، قطار)، اقامت، غذا و هزینه‌های مذهبی محاسبه می‌شود.

4. **سفرهای غیرهوایی غیرزیارتی:** هزینه حمل و نقل، اقامت، غذا و تفریحات بر اساس مقصد تعیین می‌شوند.

این روش‌ها به تخمین دقیق هزینه‌ها و تحلیل‌های مالی کمک می‌کنند.

د.

تمامی هزینه ها بر اساس تور های سایت های معتبر مانند علی بابا و اطلاعات خبرگزاری ها و همچنین قیمت بلیط های هواپیما و قطار و هزینه های ماشین شخصی بدست اورده شده همچنین از AI هم کمک گرفته شده

محاسبه هزینه سفر های هر شخص:(ریال)

- هوایی زیارتی: 250,000,000
- هوایی غیر زیارتی : 300,000,000
- زمینی زیارتی : 80,000,000
- زمینی غیر زیارتی: 100,000,000

In [15]:
```
df['Total_trip_cost'] = (df['TripCountAirNonPilgrimage_95to99'].fillna(0) * 300000
                         df['TripCountAirPilgrimage_95to99'].fillna(0) * 250000000
                         df['TripCountNonAirNonPilgrimage_95to99'].fillna(0) * 100
                        df['TripCountNonAirPilgrimage_95to99'].fillna(0) * 8000000
```

In [16]:
```
df.head()
```

Out[16]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_shedat	Da
0	717641288407	327754746508	2	9.0	NaN	NaN	NaN	
1	84729392521	84729392521	2	30.0	NaN	NaN	NaN	
2	480898026296	480898026296	1	33.0	NaN	NaN	NaN	
3	72903495587	824385263640	2	54.0	NaN	NaN	NaN	
4	597960900957	493995376717	2	35.0	NaN	NaN	NaN	

5 rows × 50 columns

2- دارایی های ثابت(Fixed_Asset)

In [17]:
```
columns_to_fill = ['CarsPrice', 'Daramad', 'MandehAkhar_1399', 'MandehAkhar_1400',

for col in columns_to_fill:
    df[col] = df[col].fillna(0)
```

In [18]:
```
df['Fixed_Asset'] = (df['CarsPrice'] + df['Bourse_NetPortfoValue'] + df['Daramad']
```

In [19]:
```
df.head()
```

Out[19]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_shedat	Da
0	717641288407	327754746508	2	9.0	NaN	NaN	NaN	
1	84729392521	84729392521	2	30.0	NaN	NaN	NaN	
2	480898026296	480898026296	1	33.0	NaN	NaN	NaN	
3	72903495587	824385263640	2	54.0	NaN	NaN	NaN	
4	597960900957	493995376717	2	35.0	NaN	NaN	NaN	

5 rows × 51 columns

3- ثروت های نقدی جاری(Cash_Wealth)

In [20]:
```
columns_to_fill = ['SatnaPerMonth_1402', 'SatnaPerMonth_1401' , 'PayaPerMonth_1402'
'CardPerMonth_1402', 'CardPerMonth_1401', 'CardPerMonth_1400', 'CardPerMonth_1399',

for col in columns_to_fill:
    df[col] = df[col].fillna(0)
```

In [21]:
```
df['Cash_Wealth'] = (df['CardPerMonth_1398'] + df['CardPerMonth_1399'] + df['CardPe
+ df['CardBeCardPerMonth_1401']+ df['CardBeCardPerMonth_1402'] + df['PayaPerMonth_1
```

```
+ df['SatnaPerMonth_1402'])
```

In [22]:
```
df
```

Out[22]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_she
0	717641288407	327754746508	2	9.0	NaN	NaN	N
1	84729392521	84729392521	2	30.0	NaN	NaN	N
2	480898026296	480898026296	1	33.0	NaN	NaN	N
3	72903495587	824385263640	2	54.0	NaN	NaN	N
4	597960900957	493995376717	2	35.0	NaN	NaN	N
...	...	...	...	...	...	...	
1697811	53326817442	557444773010	1	12.0	NaN	NaN	N
1697812	208062992297	408375687751	2	33.0	NaN	NaN	N
1697813	360855746695	679652402477	1	41.0	NaN	NaN	N
1697814	513741239072	218828740607	2	54.0	NaN	NaN	N
1697815	231940330996	231940330996	1	27.0	NaN	NaN	N

1697816 rows × 52 columns

ایجاد ستون 'Servat'

In [23]:
```
df['Servat'] = (df['Cash_Wealth'] + df ['Fixed_Asset'] + df['Total_trip_cost'])
```

In [24]:
```
df
```

Out[24]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	Malool_she
0	717641288407	327754746508	2	9.0	NaN	NaN	N
1	84729392521	84729392521	2	30.0	NaN	NaN	N
2	480898026296	480898026296	1	33.0	NaN	NaN	N
3	72903495587	824385263640	2	54.0	NaN	NaN	N
4	597960900957	493995376717	2	35.0	NaN	NaN	N
...	...	...	...	...	...	...	
1697811	53326817442	557444773010	1	12.0	NaN	NaN	N
1697812	208062992297	408375687751	2	33.0	NaN	NaN	N
1697813	360855746695	679652402477	1	41.0	NaN	NaN	N
1697814	513741239072	218828740607	2	54.0	NaN	NaN	N
1697815	231940330996	231940330996	1	27.0	NaN	NaN	N

1697816 rows × 53 columns

In [25]:
```
df.describe()
```

Out[25]:

	id	Parent_Id	GenderId	Age	ISBimarKhas	IsMalool	M
count	1.697816e+06	1.697816e+06	1.697816e+06	1.696590e+06	13331.0	28419.0	
mean	4.361329e+11	4.332060e+11	1.494946e+00	3.408394e+01	1.0	1.0	
std	2.524222e+11	2.500585e+11	4.999746e-01	2.031489e+01	0.0	0.0	
min	2.743800e+04	2.599360e+05	1.000000e+00	0.000000e+00	1.0	1.0	
25%	2.178905e+11	2.167209e+11	1.000000e+00	1.700000e+01	1.0	1.0	
50%	4.355807e+11	4.330983e+11	1.000000e+00	3.400000e+01	1.0	1.0	
75%	6.538524e+11	6.498149e+11	2.000000e+00	4.800000e+01	1.0	1.0	
max	9.999962e+11	9.995183e+11	2.000000e+00	9.900000e+01	1.0	1.0	

8 rows × 50 columns

ثروت خانوار ها(سرپرست خانواده)

In [26]:
```
#جدا کردن فیچر های مورد استفاده برای مصور سازی داده ها
parent_features = df.groupby('Parent_Id').agg({
```

```
    'Age': 'first',
    'GenderId': 'first',
    'SabteAhval_provincename': 'first',
    'SabteAhval_countyname': 'first',
    'Decile': 'first',
    'Percentile': 'first',
    'Yaraneh': 'first'
}).reset_index()
```

In [27]:
```
parent_features
```

Out[27]:

	Parent_Id	Age	GenderId	SabteAhval_provincename	SabteAhval_countyname
0	259936	33.0	1	تهران	تهران
1	2074978	32.0	1	سمنان	شاهرود
2	2256993	27.0	1	مازندران	بهشهر
3	3957230	38.0	1	لرستان	کوهدشت
4	4588065	4.0	1	تهران	تهران
...	...	...	...	...	...
595683	997477310083	41.0	1	None	None
595684	998551911230	95.0	1	None	None
595685	999165154002	95.0	1	None	None
595686	999363329496	51.0	1	None	None
595687	999518333981	5.0	2	مرکزی	محلات

595688 rows × 8 columns

In [28]:
```
df['Age'] = df.Age.astype('str')
df['id'] = df.id.astype('str')
df['GenderId'] = df.GenderId.astype('str')
```

In [29]:
```
# پیمایش بر روی سرپرست های خانواده
df_family = df.pivot_table(index='Parent_Id',
                           values= ['id', 'Age','GenderId', 'Servat'],
                           aggfunc= {'id':
                                     [np.count_nonzero, lambda x:','.join(x)],
                                     'Age':
                                     [lambda x:','.join(x)
                                      #, lambda x:set(x)
                                     ],
                                     'GenderId':
                                     [lambda x:','.join(x)],
                                     'Servat' : "sum"

                                    }
```

```
).reset_index()
```

In [30]:
```
df_family
```

Out[30]:

	Parent_Id	Age	GenderId	Servat	
		<lambda>	<lambda>	sum	
0	259936	33.0,30.0	1,2	1.462075e+10	
1	2074978	32.0,62.0,36.0,68.0	1,2,1,1	6.446604e+09	585352131467,818175
2	2256993	27.0,24.0	1,2	1.053242e+10	
3	3957230	38.0,48.0	1,2	1.951770e+10	
4	4588065	4.0,39.0,14.0,7.0,39.0	1,2,1,1,1	2.429623e+10	400325360511,4943377
...	...	...	...	...	
595683	997477310083	41.0	1	1.759142e+09	
595684	998551911230	95.0	1	0.000000e+00	
595685	999165154002	95.0	1	7.035225e+07	
595686	999363329496	51.0	1	1.359587e+07	
595687	999518333981	5.0,32.0	2,1	5.123736e+09	

595688 rows × 6 columns

In [31]:
```
df_family.columns = ['Parent_Id', 'Age_family', 'GenderId_family', 'Servat', 'id_fa
```

In [32]:
```
df_family
```

Out[32]:

	Parent_Id	Age_family	GenderId_family	Servat	
0	259936	33.0,30.0	1,2	1.462075e+10	
1	2074978	32.0,62.0,36.0,68.0	1,2,1,1	6.446604e+09	585352131467,8
2	2256993	27.0,24.0	1,2	1.053242e+10	
3	3957230	38.0,48.0	1,2	1.951770e+10	
4	4588065	4.0,39.0,14.0,7.0,39.0	1,2,1,1,1	2.429623e+10	400325360511,49
...	...	...	...	...	
595683	997477310083	41.0	1	1.759142e+09	
595684	998551911230	95.0	1	0.000000e+00	
595685	999165154002	95.0	1	7.035225e+07	
595686	999363329496	51.0	1	1.359587e+07	
595687	999518333981	5.0,32.0	2,1	5.123736e+09	

595688 rows × 6 columns

- پیوند دو داده جدا شده به یکدیگر

In [33]:
```
df_family = df_family.merge(parent_features, on='Parent_Id', how='left')
```

In [34]:
```
df_family
```

Out[34]:

	Parent_Id	Age_family	GenderId_family	Servat	
0	259936	33.0,30.0	1,2	1.462075e+10	
1	2074978	32.0,62.0,36.0,68.0	1,2,1,1	6.446604e+09	585352131467,8
2	2256993	27.0,24.0	1,2	1.053242e+10	
3	3957230	38.0,48.0	1,2	1.951770e+10	
4	4588065	4.0,39.0,14.0,7.0,39.0	1,2,1,1,1	2.429623e+10	400325360511,49
...	...	...	...	...	
595683	997477310083	41.0	1	1.759142e+09	
595684	998551911230	95.0	1	0.000000e+00	
595685	999165154002	95.0	1	7.035225e+07	
595686	999363329496	51.0	1	1.359587e+07	
595687	999518333981	5.0,32.0	2,1	5.123736e+09	

595688 rows × 13 columns

ايجاد ستون 'Servat_Decile'

In [35]:
```
percentile_80 = df_family['Servat'].quantile(0.8)
percentile_90 = df_family['Servat'].quantile(0.9)
```

In [36]:
```
def servat_category(row):
    if row['Servat'] > percentile_90:
        return 'very high wealth'
    elif row['Servat'] > percentile_80:
        return 'High Wealth'
    else:
        return 'None'
```

In [37]:
```
df_family['Servat_Decile'] = df_family.apply(servat_category, axis = 1)
```

In [38]:
```
df_family.head()
```

Out[38]:

	Parent_Id	Age_family	GenderId_family	Servat	
0	259936	33.0,30.0	1,2	1.462075e+10	
1	2074978	32.0,62.0,36.0,68.0	1,2,1,1	6.446604e+09	585352131467,8181752989
2	2256993	27.0,24.0	1,2	1.053242e+10	
3	3957230	38.0,48.0	1,2	1.951770e+10	
4	4588065	4.0,39.0,14.0,7.0,39.0	1,2,1,1,1	2.429623e+10	400325360511,49433777191

ایجاد ستون 'Tax_fruad'

In [39]:
```
df_family['Tax_fruad'] = ((df_family['Yaraneh'] == True) & (df_family['Servat_Decil
```

In [40]:
```
df_family['Tax_fruad'] = df_family['Tax_fruad'].map({'True': True, 'False': False})
```

In [41]:
```
df_family['Tax_fruad'] = df_family['Tax_fruad'].astype(int)
```

In [42]:
```
df_family
```

Out[42]:

	Parent_Id	Age_family	GenderId_family	Servat	
0	259936	33.0,30.0	1,2	1.462075e+10	
1	2074978	32.0,62.0,36.0,68.0	1,2,1,1	6.446604e+09	585352131467,8
2	2256993	27.0,24.0	1,2	1.053242e+10	
3	3957230	38.0,48.0	1,2	1.951770e+10	
4	4588065	4.0,39.0,14.0,7.0,39.0	1,2,1,1,1	2.429623e+10	400325360511,49
...	...	...	...	...	
595683	997477310083	41.0	1	1.759142e+09	
595684	998551911230	95.0	1	0.000000e+00	
595685	999165154002	95.0	1	7.035225e+07	
595686	999363329496	51.0	1	1.359587e+07	
595687	999518333981	5.0,32.0	2,1	5.123736e+09	

595688 rows × 15 columns

In [43]:
```
df_family.describe()
```

Out[43]:

	Parent_Id	Servat	family_member	Age	GenderId	
count	5.956880e+05	5.956880e+05	595688.000000	594753.000000	595688.000000	595688.(
mean	4.334345e+11	2.976666e+10	2.850177	38.249359	1.510991	5.3
std	2.501756e+11	6.489989e+10	1.409557	21.078369	0.499880	2.9
min	2.599360e+05	0.000000e+00	1.000000	0.000000	1.000000	1.(
25%	2.167361e+11	3.820148e+09	2.000000	23.000000	1.000000	3.(
50%	4.334774e+11	1.036284e+10	3.000000	37.000000	2.000000	5.(
75%	6.501855e+11	2.666200e+10	4.000000	53.000000	2.000000	8.(
max	9.995183e+11	6.810960e+12	21.000000	99.000000	2.000000	10.(

کیس های مشکوک به فرار مالیاتی

In [44]:
```
df_family.Tax_fruad.sum()
```

Out[44]: 869

- جدا کردن داده های مثبت فرار مالیاتی

In [45]:
```
tax_evasion_df = df_family[df_family['Tax_fruad'] == 1]
```

In [46]:
```
tax_evasion_df
```

Out[46]:

	Parent_Id	Age_family	GenderId_family	Servat	
9	13211647	9.0,11.0,43.0,45.0,2.0	1,1,2,1,2	4.894230e+10	76959923
846	1161467736	38.0	1	6.019670e+10	
924	1287000869	2.0,44.0,6.0,39.0	1,1,2,2	3.710853e+10	938333458
1269	1801451991	46.0,18.0,16.0	2,1,1	6.246412e+10	
1961	2805295095	67.0,25.0,28.0,74.0	2,1,1,1	4.806941e+10	7028515
...	...	...	...	...	
590170	858200153987	69.0,34.0,45.0,40.0,75.0,34.0	2,1,1,1,1,2	3.639212e+10	19733012
590389	858501089007	8.0,32.0,32.0	1,2,1	4.885761e+10	
590768	859079470759	6.0,30.0,36.0	2,2,1	4.182937e+10	
591906	860723643856	25.0,16.0,1.0	1,2,1	4.549657e+10	
594184	864069272740	63.0,55.0,33.0,30.0,25.0	1,2,2,1,2	3.802945e+10	82512881

869 rows × 15 columns

In [47]:
```
tax_evasion_df.describe()
```

Out[47]:

	Parent_Id	Servat	family_member	Age	GenderId	Decile	F
count	8.690000e+02	8.690000e+02	869.000000	868.000000	869.000000	869.000000	86
mean	4.416246e+11	4.659394e+10	3.546605	34.987327	1.509781	2.438435	2
std	2.529745e+11	9.868800e+09	1.762174	21.189453	0.500192	0.658126	
min	1.321165e+07	3.438804e+10	1.000000	0.000000	1.000000	1.000000	
25%	2.158909e+11	3.835714e+10	2.000000	17.000000	1.000000	2.000000	1
50%	4.555345e+11	4.362366e+10	4.000000	35.000000	2.000000	3.000000	2
75%	6.670485e+11	5.512508e+10	5.000000	49.000000	2.000000	3.000000	2
max	8.640693e+11	6.861888e+10	21.000000	93.000000	2.000000	3.000000	3

مصور سازی داده ها

- بر حسب جنسیت
- بر حسب استان

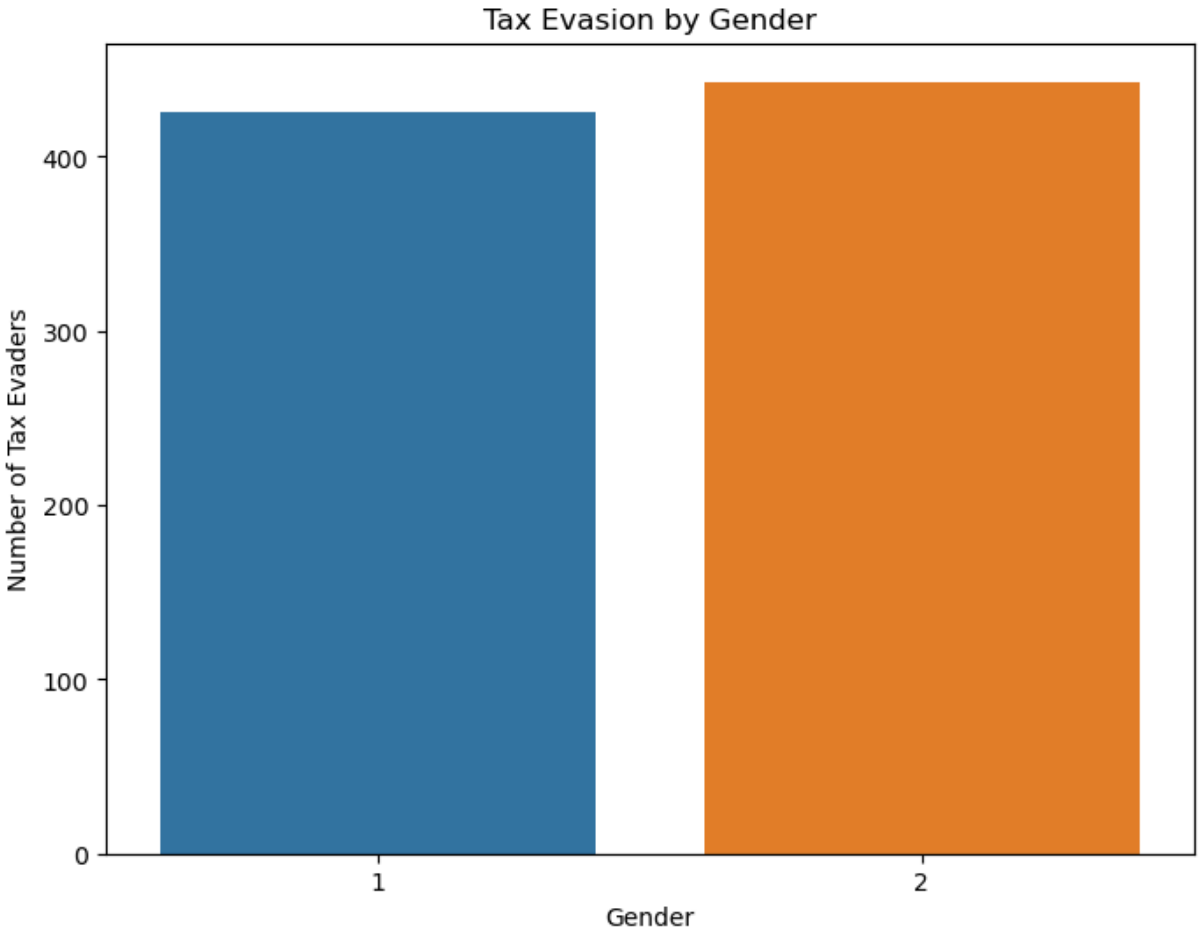
- بر حسب دهه تولد
- بر حسب سن

In [48]:
```
tax_evasion_df.dtypes
```

Out[48]:
```
Parent_Id                  int64
Age_family                object
GenderId_family           object
Servat                   float64
id_family                 object
family_member              int64
Age                      float64
GenderId                   int64
SabteAhval_provincename   object
SabteAhval_countyname     object
Decile                     int64
Percentile                 int64
Yaraneh                     bool
Servat_Decile             object
Tax_fruad                  int32
dtype: object
```

- مصور سازی داده ها بر حسب جنسیت

In [49]:
```
plt.figure(figsize=(8, 6))
sns.countplot(x='GenderId', data=tax_evasion_df)

plt.title('Tax Evasion by Gender')
plt.xlabel('Gender')
plt.ylabel('Number of Tax Evaders')
plt.show()
```



- مصور سازی داده ها بر حسب استان محل اقامت

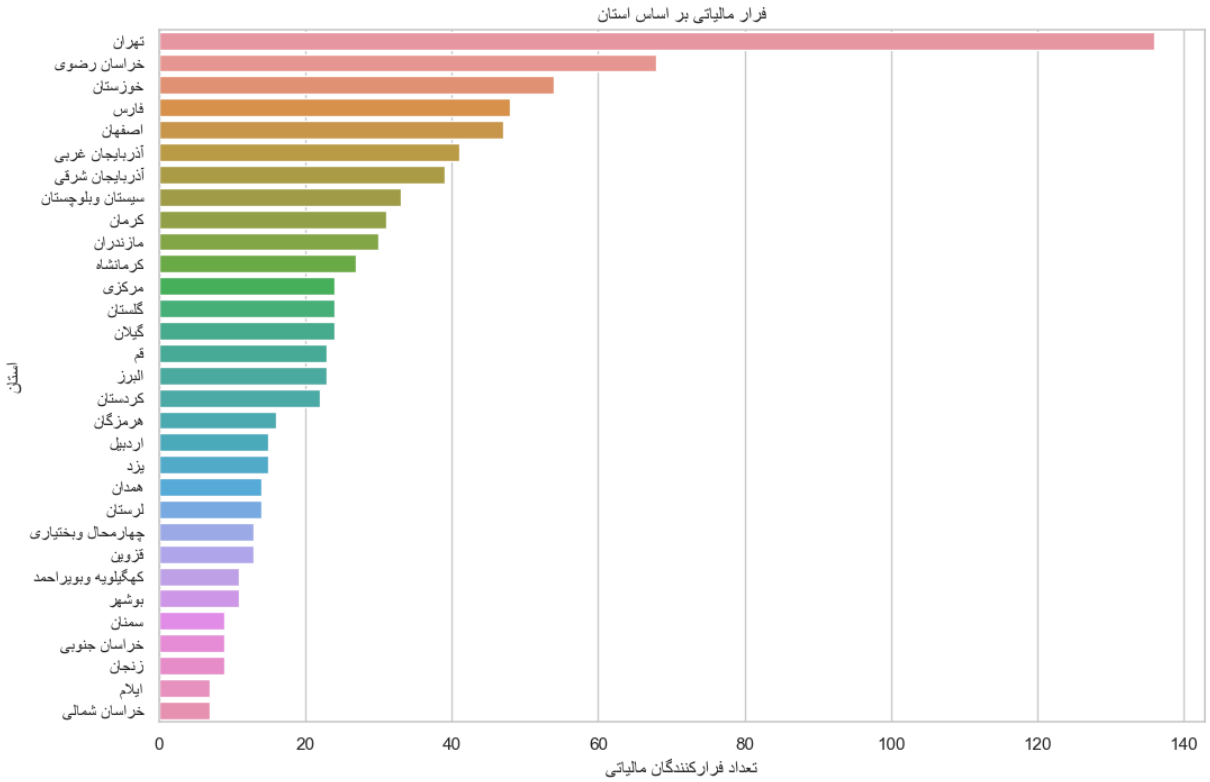
برطرف کردن باگ نخواندن استانها به زبان مرود استفاده در دیتاست(فارسی)

In [50]:
```
import arabic_reshaper
import bidi.algorithm
from matplotlib import font_manager

def text_fa(x):
    x = arabic_reshaper.reshape(x)
    x = bidi.algorithm.get_display(x)
    return x


font = {'family' : 'B Mitra', 'size' : 21}
plt.rc('font', **font)
```

In [51]:
```
sns.set(style="whitegrid")
plt.figure(figsize=(12, 8))

# نمودار میله‌ای
```

```
plt.figure(figsize=(12, 8))

sns.countplot(y='SabteAhval_provincename', data=tax_evasion_df, order=tax_evasion_d

# اعمال تابع text_fa
plt.yticks(ticks=range(len(tax_evasion_df['SabteAhval_provincename'].value_counts()

plt.title(text_fa('فرار مالیاتی بر اساس استان'))
plt.xlabel(text_fa('تعداد فرارکنندگان مالیاتی'))
plt.ylabel(text_fa('استان'))
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```

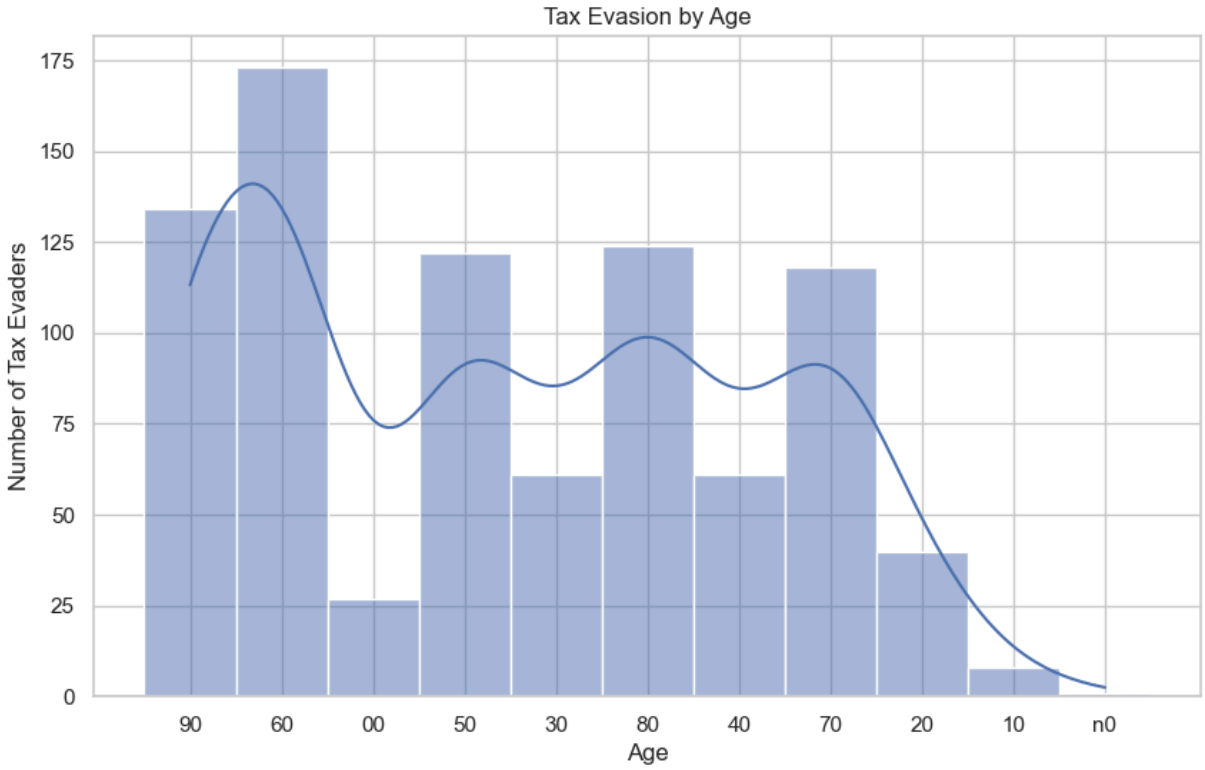


- مصور سازی داده ها بر حسب دهه تولد

In [52]:

```
# نمودار هیستوگرام بر حسب دهه تولد
tax_evasion_df['Brith_year'] = 1402 - tax_evasion_df['Age']
tax_evasion_df['Brith_decade'] = tax_evasion_df.Brith_year.astype('str').str[2:3] +

plt.figure(figsize=(10, 6))
sns.histplot(tax_evasion_df['Brith_decade'], bins=20, kde=True)

plt.title('Tax Evasion by Age')
plt.xlabel('Age')
plt.ylabel('Number of Tax Evaders')
plt.show()
```

```
C:\Users\ZBook 15 G3\AppData\Local\Temp\ipykernel_7748\1744505979.py:2: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  tax_evasion_df['Brith_year'] = 1402 - tax_evasion_df['Age']
C:\Users\ZBook 15 G3\AppData\Local\Temp\ipykernel_7748\1744505979.py:3: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  tax_evasion_df['Brith_decade'] = tax_evasion_df.Brith_year.astype('str').str[2:3]
+ '0'
C:\Users\ZBook 15 G3\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWar
ning: use_inf_as_na option is deprecated and will be removed in a future version. Co
nvert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

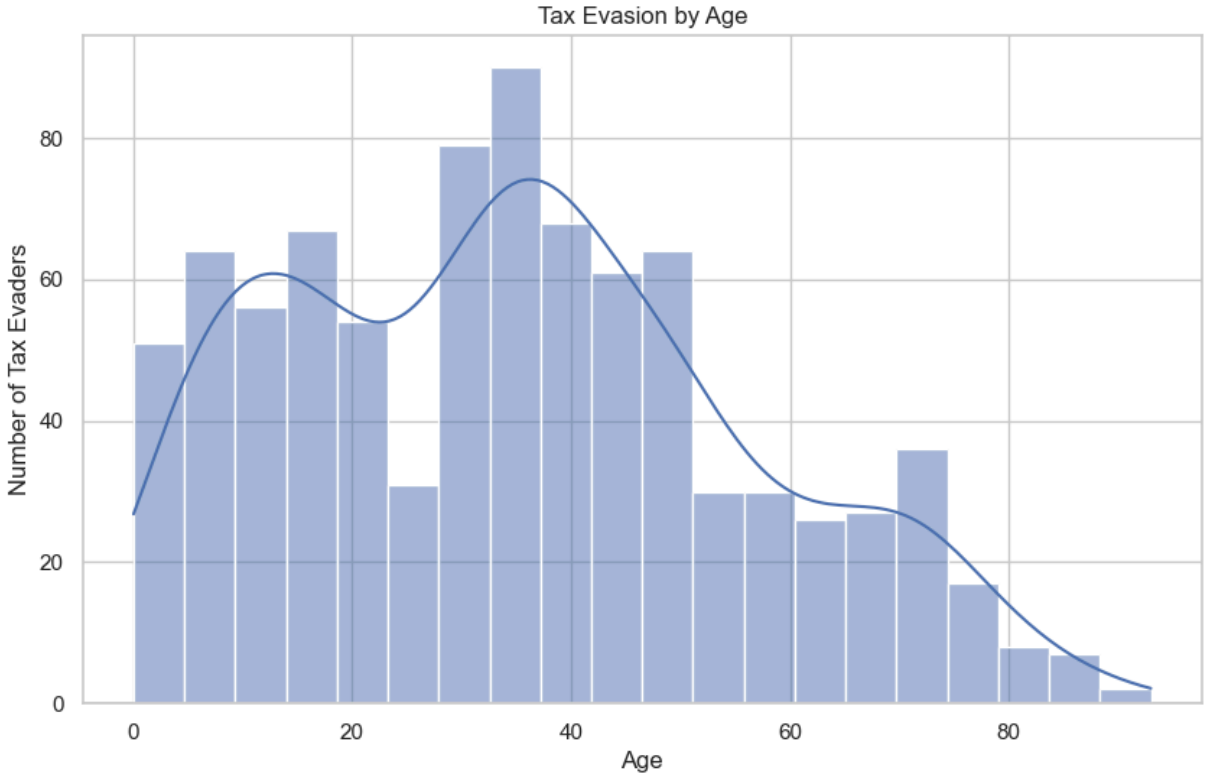


- مصور سازی داده ها بر حسب سن

In [53]:
```
# نمودار هیستوگرام برای سن

plt.figure(figsize=(10, 6))
sns.histplot(tax_evasion_df['Age'], bins=20, kde=True)
```

```
plt.title('Tax Evasion by Age')
plt.xlabel('Age')
plt.ylabel('Number of Tax Evaders')
plt.show()
```

```
C:\Users\ZBook 15 G3\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



جمع بندی

در تحلیل این پروژه با توجه به اینکه داده های موجود در دیتاست تکمیل نبود و کاستی های زیادی داشت همچنین عدم وجود داده های کلیدی تر برای بدست اوردن دقیق تر ستون ثروت مانند ملک یا املاک به اسم هر شخص باعث شد که نتوانیم تحلیل دقیق تری داشته باشیم هرچند فیچر های نا مشخص هر شخص را میتوان با روش ها جایگزینی حدس زد که بتوان مدل خود را دقیق تر کرد.

با این حال بر اساس داده های موجود و تحلیل انجام شده نرخ فرار مالیاتی سرپرست های خانوار 0.14% محاسبه شده

پیشنهادات:

- جمع آوری داده های تکمیلی
- مدیریت داده ها مفقود
- تحلیل بیشتر

جمع آوری داده های بیشتر: اضافه کردن اطلاعات مربوط به املاک و دارایی‌های غیرمنقول به داده‌ها می‌تواند به دقت بیشتری در محاسبه ثروت و شناسایی فرار مالیاتی منجر شود.

مدیریت داده های مفقود: استفاده از روش‌های پیشرفته‌تر برای جایگزینی مقادیر NaN و بررسی تأثیر این داده‌ها بر نتایج نهایی.

تحلیل‌های بیشتر: بررسی دقیق‌تر رابطه بین ویژگی‌های اقتصادی خانوار و فرار مالیاتی و استفاده از مدل‌های پیشرفته ماشین لرنینگ برای پیش‌بینی و شناسایی الگوهای مشکوک.

نتیجه گیری:

با وجود محدودیت‌ها در داده‌ها، تحلیل‌های انجام‌شده توسط بنده نشان داد که نرخ فرار مالیاتی بسیار پایین بوده و این مسئله نشان‌دهنده کنترل نسبتاً مناسب در سیستم مالیاتی است. با این حال، بهبود کیفیت داده‌ها و تکمیل اطلاعات می‌تواند به شناسایی دقیق‌تر موارد فرار مالیاتی کمک کند.

About

Name: hooman poursartip

gmail: (homun.poursartip@gmail.com)