In [1]:

```
import os
import joblib
import numpy as np
import pandas as pd
from sklearn.tree import _tree
from sklearn.metrics import roc_auc_score
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection  import train_test_split
```

In [2]:

```
sample=pd.read_csv('C:\\Users\\lalit\\Downloads\\sampled_data2.csv')
sample
```

Out[2]:

	step	customer	age	gender	merchant	category	amount	fraud
0	2021-07-16	C1326593353	36to45	Male	M1823072687	Transportation	15.78	0
1	2021-08-22	C483912131	26to35	Female	M1198415165	Beauty&Wellness	608.32	1
2	2021-06-03	C1681589600	26to35	Female	M348934600	Transportation	43.76	0
3	2021-07-01	C1251749294	36to45	Female	M1823072687	Transportation	41.57	0
4	2021-03-06	C1705346216	36to45	Male	M348934600	Transportation	42.97	0
...	...	...	...	...	...	...	...	...
25195	2021-08-22	C1480193980	46to55	Female	M1823072687	Transportation	32.07	0
25196	2021-08-14	C207380283	36to45	Female	M480139044	Health	373.27	1
25197	2021-07-18	C183270065	lt18	Female	M1823072687	Transportation	2.50	0
25198	2021-07-23	C149698808	26to35	Female	M1823072687	Transportation	5.38	0
25199	2021-07-16	C684162308	56to65	Female	M840466850	Technology	93.89	1

25200 rows × 8 columns

In [3]:

```
print(sample.shape)
print(sample.size)
```

```
(25200, 8)
201600
```

In [4]:

```
entity_column='customer'
target_column='fraud'
datetime_columns='step'
```

In [5]:

```
if entity_column=='':
    sample['entity_column']=sample.index.tolist()
    entity_column='entity_column'
if datetime_columns==" ":
    sample['datetime_columns']=sample.index.tolist()
    datetime_columns='datetime_columns'
numerical_columns=sample._get_numeric_data().values.tolist()
print(numerical_columns)
```

```
[[15.78, 0.0], [608.32, 1.0], [43.76, 0.0], [41.57, 0.0], [42.97, 0.0], [3
2.61, 0.0], [410.67, 1.0], [16.45, 1.0], [300.62, 1.0], [47.47, 0.0], [39
0.48, 1.0], [30.61, 0.0], [67.28, 0.0], [5.54, 0.0], [760.26, 1.0], [135.0
4, 0.0], [41.46, 0.0], [27.39, 0.0], [5.52, 0.0], [423.02, 1.0], [51.26,
0.0], [45.62, 0.0], [69.37, 0.0], [47.41, 0.0], [119.15, 1.0], [9.27, 0.
0], [33.31, 0.0], [22.6, 0.0], [70.0, 0.0], [1242.08, 1.0], [0.68, 0.0],
[18.34, 0.0], [34.42, 0.0], [33.3, 0.0], [20.41, 0.0], [297.6, 1.0], [12.9
2, 0.0], [267.3, 1.0], [23.49, 0.0], [68.31, 0.0], [155.95, 1.0], [49.83,
0.0], [3.96, 0.0], [20.62, 0.0], [152.55, 1.0], [21.44, 0.0], [143.62, 0.
0], [49.78, 0.0], [11.58, 0.0], [11.37, 0.0], [249.43, 1.0], [37.9, 0.0],
[512.51, 1.0], [28.27, 0.0], [27.82, 0.0], [204.29, 1.0], [14.84, 0.0], [1
88.92, 1.0], [1.14, 0.0], [166.08, 1.0], [21.18, 0.0], [17.5, 0.0], [15.9
8, 0.0], [26.35, 0.0], [310.76, 1.0], [305.05, 1.0], [362.29, 1.0], [320.7
5, 1.0], [426.13, 1.0], [205.15, 1.0], [635.36, 1.0], [303.91, 1.0], [17.4
7, 0.0], [27.38, 0.0], [224.41, 1.0], [18.71, 0.0], [59.13, 0.0], [199.75,
1.0], [167.62, 1.0], [475.95, 1.0], [4775.72, 1.0], [25.85, 0.0], [4.52,
0.0], [55.98, 0.0], [487.12, 1.0], [16.19, 0.0], [25.14, 0.0], [427.78, 1.
0], [25.48, 0.0], [24.7, 0.0], [165.11, 1.0], [28.0, 0.0], [13.05, 0.0],
[24.03, 0.0], [26.75, 1.0], [10.2, 0.0], [44.64, 0.0], [35.17, 0.0], [38.9
```

In [6]:

```
X=sample.drop(columns=[entity_column,target_column,datetime_columns])
y=sample[target_column]
```

In [7]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
```

In [8]:

```
X
```

Out[8]:

	age	gender	merchant	category	amount
0	36to45	Male	M1823072687	Transportation	15.78
1	26to35	Female	M1198415165	Beauty&Wellness	608.32
2	26to35	Female	M348934600	Transportation	43.76
3	36to45	Female	M1823072687	Transportation	41.57
4	36to45	Male	M348934600	Transportation	42.97
...	...	...	...	...	...
25195	46to55	Female	M1823072687	Transportation	32.07
25196	36to45	Female	M480139044	Health	373.27
25197	lt18	Female	M1823072687	Transportation	2.50
25198	26to35	Female	M1823072687	Transportation	5.38
25199	56to65	Female	M840466850	Technology	93.89

25200 rows × 5 columns

In [9]:

```
categorical_names={}
for feature in X.columns:
    X[feature]=X[feature].fillna('')
    le=LabelEncoder()
    le.fit(X[feature])
    X[feature]=le.transform(X[feature])
    categorical_names[feature]=le.classes_
```

In [10]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.75,random_state=42,stratify=
rf=RandomForestClassifier(criterion='entropy',class_weight='balanced',random_state=42)
rf.fit(X_train,y_train)
```

Out[10]:

```
RandomForestClassifier(class_weight='balanced', criterion='entropy',
                       random_state=42)
```

In [11]:

```
predictions=rf.predict_proba(X_test)
predictions_class=rf.predict(X_test)
```

In [12]:

```
if sample[target_column].nunique()>2:
    rules_score=roc_auc_score(y_test,predictions[:,1],multi_class='ovo')
else:
    rules_score=roc_auc_score(y_test,predictions[:,1])
print(rules_score)
```

```
0.9960853703703704
```

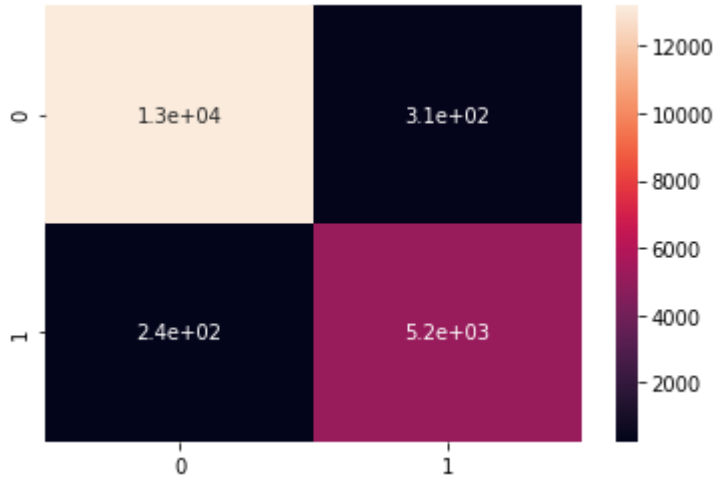
In [17]:

```
from sklearn.metrics import confusion_matrix
cf_matrix = confusion_matrix(y_test, predictions_class)
print(cf_matrix)

import seaborn as sns
sns.heatmap(cf_matrix, annot=True)
```

```
[[13193   307]
 [  243  5157]]
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e1e22c2948>
```



In [13]:

```
# performing EDA on the given data
import matplotlib.pyplot as plt
```

In [19]:

```
sample.describe(include='all')
```

Out[19]:

	step	customer	age	gender	merchant	category	amount	
count	25200	25200	25200	25200	25200	25200	25200.000000	25200.
unique	180	4061	8	4	48	14	NaN	
top	2021-06-18	C1350963410	26to35	Female	M1823072687	Transportation	NaN	
freq	177	147	7984	14596	9192	15925	NaN	
mean	NaN	NaN	NaN	NaN	NaN	NaN	174.212506	0.
std	NaN	NaN	NaN	NaN	NaN	NaN	501.116240	0.
min	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	0.
25%	NaN	NaN	NaN	NaN	NaN	NaN	17.720000	0.
50%	NaN	NaN	NaN	NaN	NaN	NaN	36.340000	0.
75%	NaN	NaN	NaN	NaN	NaN	NaN	109.180000	1.
max	NaN	NaN	NaN	NaN	NaN	NaN	8329.960000	1.

In [20]:

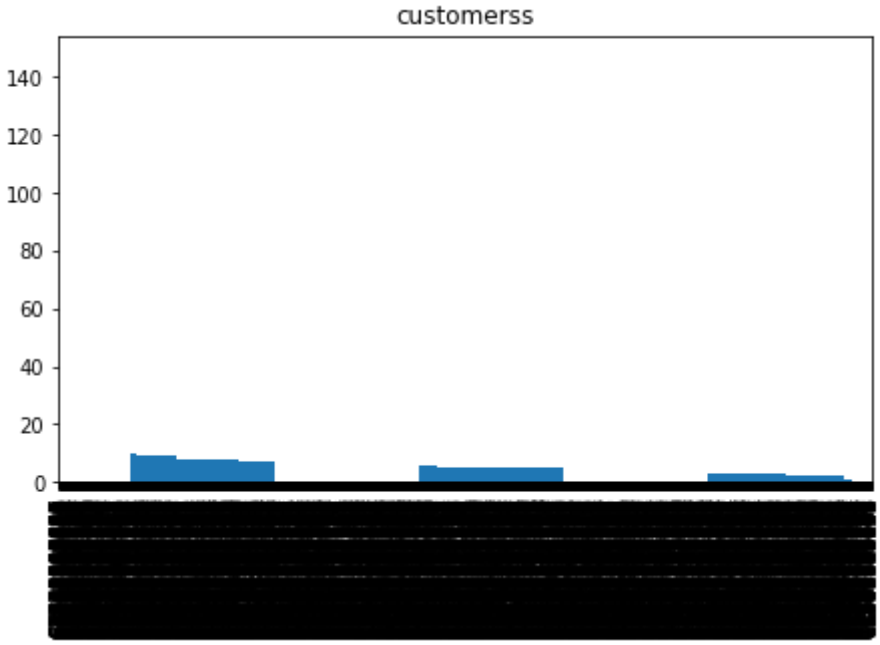
```
sample.isnull().sum()
```

Out[20]:

```
step        0
customer    0
age         0
gender      0
merchant    0
category    0
amount      0
fraud       0
dtype: int64
```

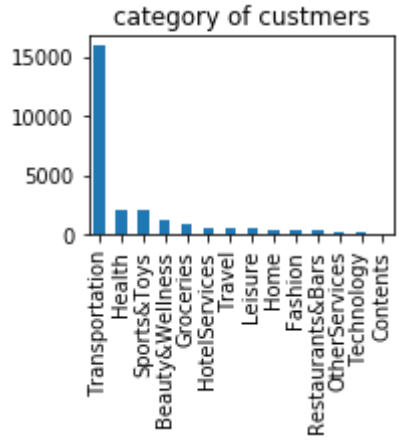
In [21]:

```
plt.subplot(221)

sample['customer'].value_counts().plot(kind='bar', title='customerss', figsize=(16,9))
plt.show()
```



In [19]:

```
plt.figsize=(50,14)
plt.subplot(222)

sample['category'].value_counts().plot(kind='bar', title='category of custmers')
plt.show() #this plot tells us a brief idea of the major number of  cistomer involved in th
```

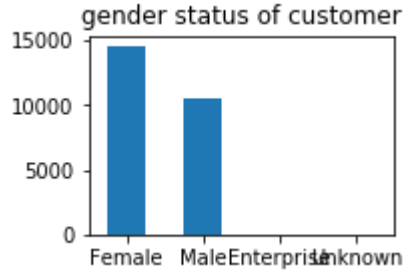


In [20]:

```
plt.subplot(223)

sample['gender'].value_counts().plot(kind='bar', title='gender status of customer')

plt.xticks(rotation=0)# this plot give us a brief identify of the gender of customer and we
```

Out[20]:

```
(array([0, 1, 2, 3]), <a list of 4 Text xticklabel objects>)
```



In [14]:

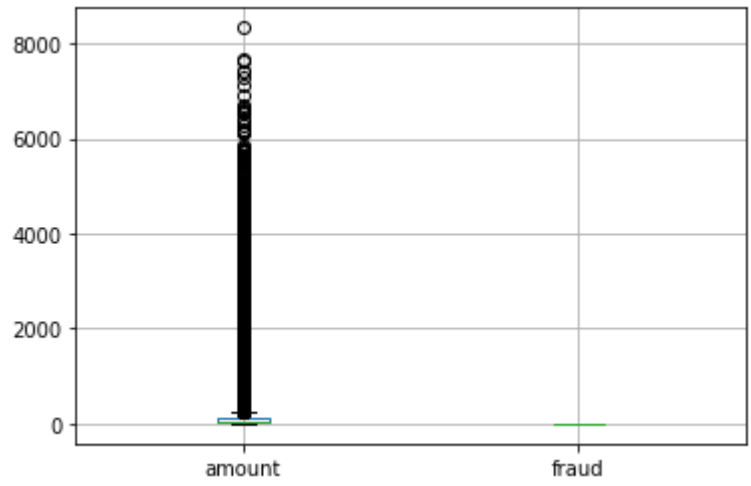
```
plt.subplot(224)
sample['fraud'].value_counts().plot(kind='bar',title='fraud rate of customer')
plt.show() # from this plot we can see that less number of people are involved in the fraud
```

In [24]:

```
sample.boxplot()
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e1ed1150c8>
```

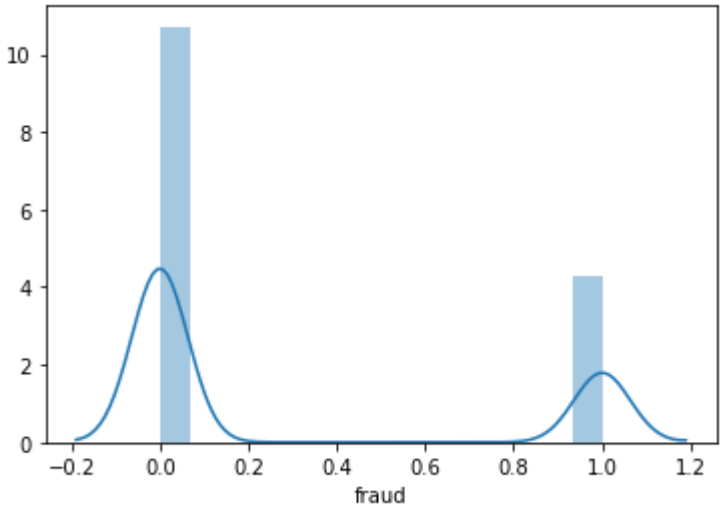


In [25]:

```
sns.distplot(sample['fraud'])
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e1eb392e48>
```



In [26]:

```
corr = sample.corr()
sns.heatmap(corr, annot=True, square=True)
plt.yticks(rotation=0)
plt.show()
```

In [28]:

```
sns.relplot(x='fraud', y='step', hue='customer', data=sample)
```

Out[28]:

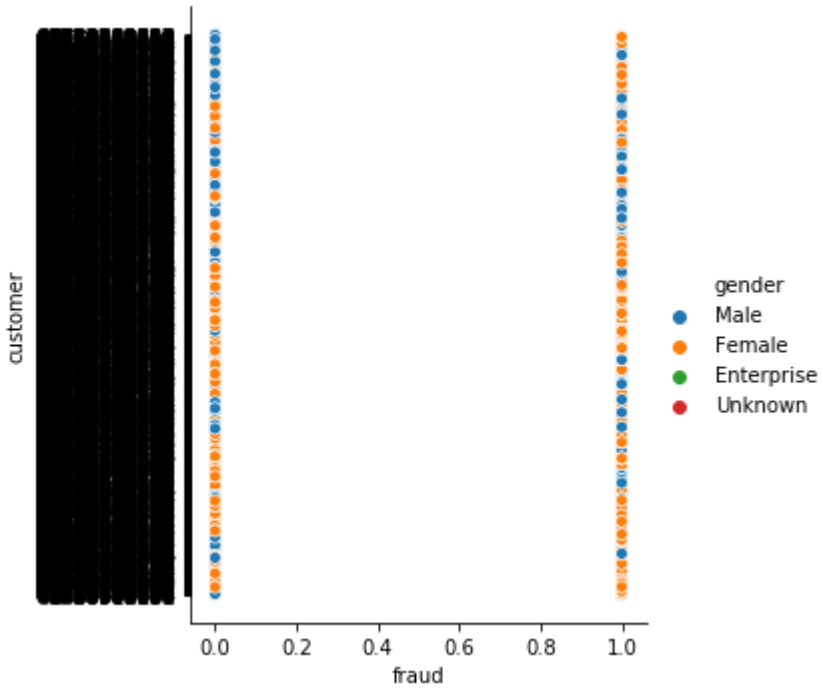
```
<seaborn.axisgrid.FacetGrid at 0x1e1ed115788>
```

In [29]:

```
sns.relplot(x='fraud', y='customer', hue='gender', data=sample)
```

Out[29]:

```
<seaborn.axisgrid.FacetGrid at 0x1e1eb35ddc8>
```

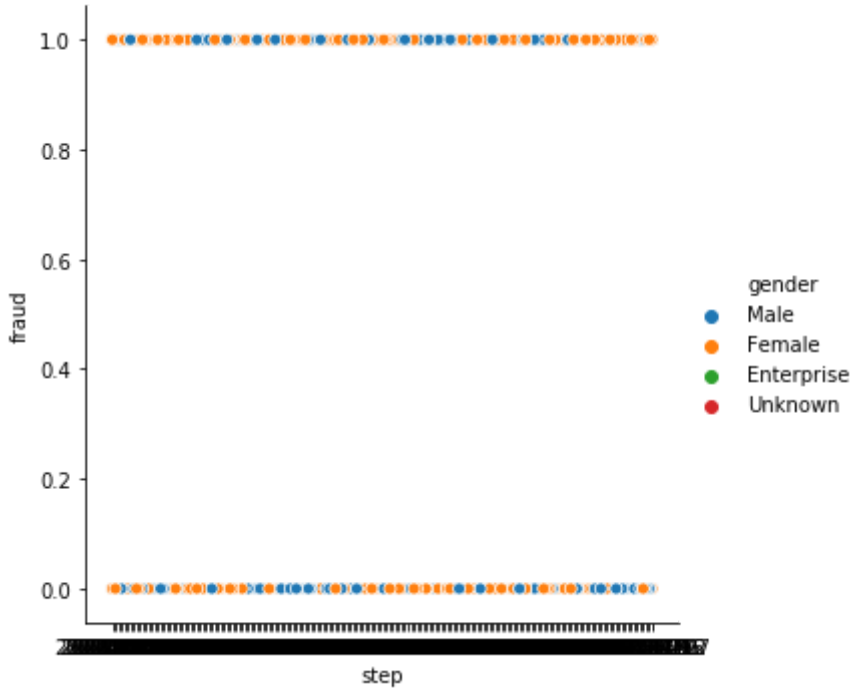


In [31]:

```
sns.relplot(x='step', y='fraud', hue='gender', data=sample)
```

Out[31]:

```
<seaborn.axisgrid.FacetGrid at 0x1e1862b4308>
```



In [32]:

```
sample
```

Out[32]:

	step	customer	age	gender	merchant	category	amount	fraud
0	2021-07-16	C1326593353	36to45	Male	M1823072687	Transportation	15.78	0
1	2021-08-22	C483912131	26to35	Female	M1198415165	Beauty&Wellness	608.32	1
2	2021-06-03	C1681589600	26to35	Female	M348934600	Transportation	43.76	0
3	2021-07-01	C1251749294	36to45	Female	M1823072687	Transportation	41.57	0
4	2021-03-06	C1705346216	36to45	Male	M348934600	Transportation	42.97	0
...	...	...	...	...	...	...	...	...
25195	2021-08-22	C1480193980	46to55	Female	M1823072687	Transportation	32.07	0
25196	2021-08-14	C207380283	36to45	Female	M480139044	Health	373.27	1
25197	2021-07-18	C183270065	lt18	Female	M1823072687	Transportation	2.50	0
25198	2021-07-23	C149698808	26to35	Female	M1823072687	Transportation	5.38	0
25199	2021-07-16	C684162308	56to65	Female	M840466850	Technology	93.89	1

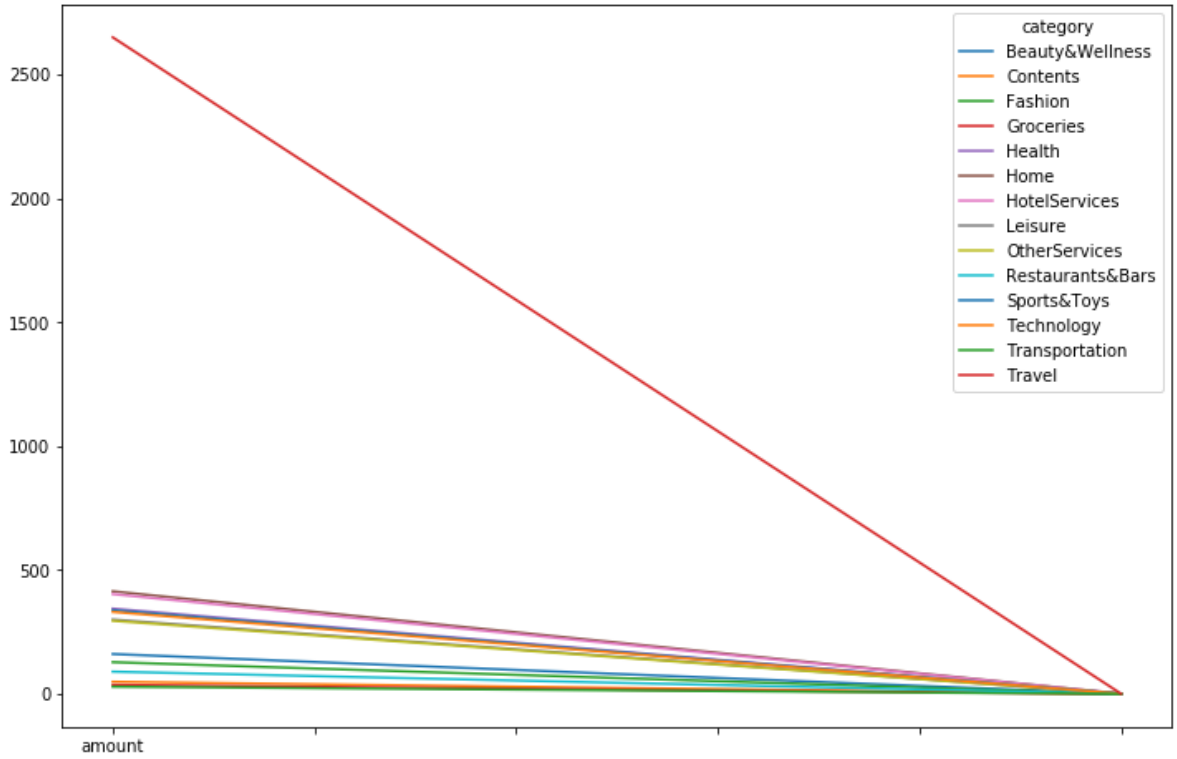
25200 rows × 8 columns

In [34]:

```
sample.groupby('category').mean().T.plot(figsize=(12,8))  # we are checking for which all i
```

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e187ebbcc8>
```

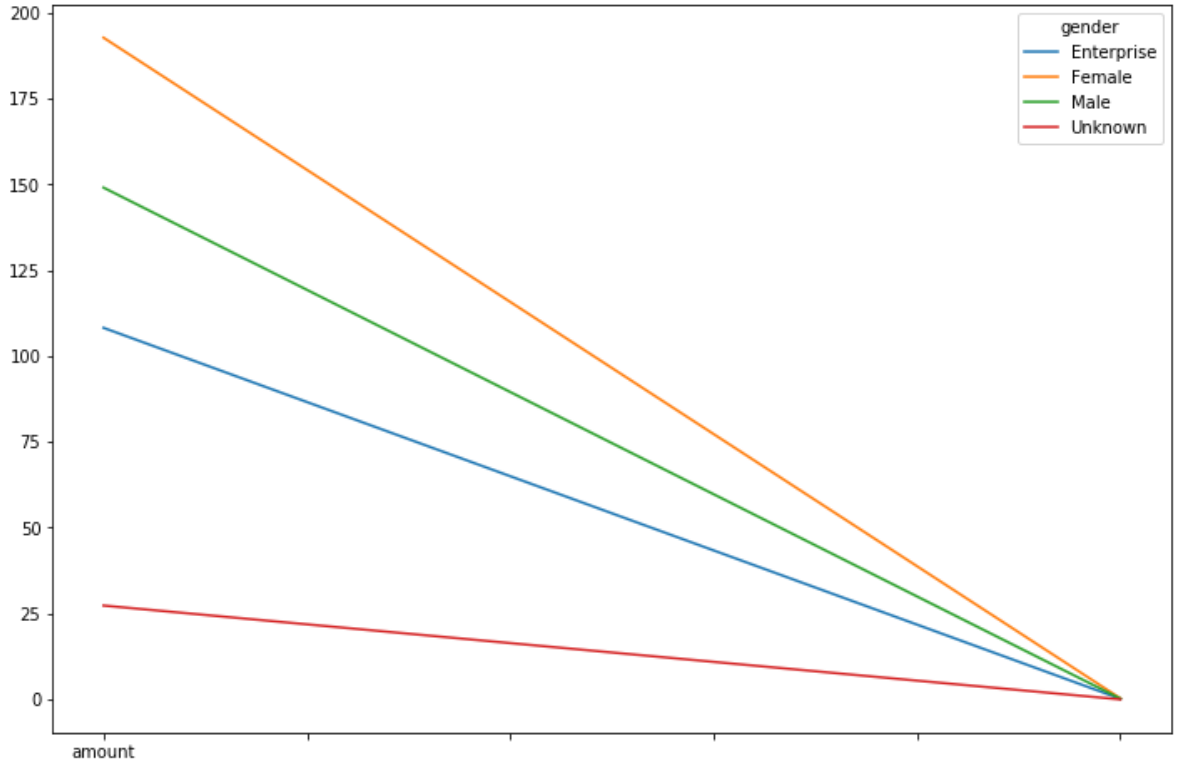


In [35]:

```
sample.groupby('gender').mean().T.plot(figsize=(12,8))  # In this we are checking the gende
```

Out[35]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e187eb8388>
```



In [22]:

```
sample.columns
```

Out[22]:

```
Index(['step', 'customer', 'age', 'gender', 'merchant', 'category', 'amoun
t',
       'fraud'],
      dtype='object')
```

In [23]:

```
sample.info()
```

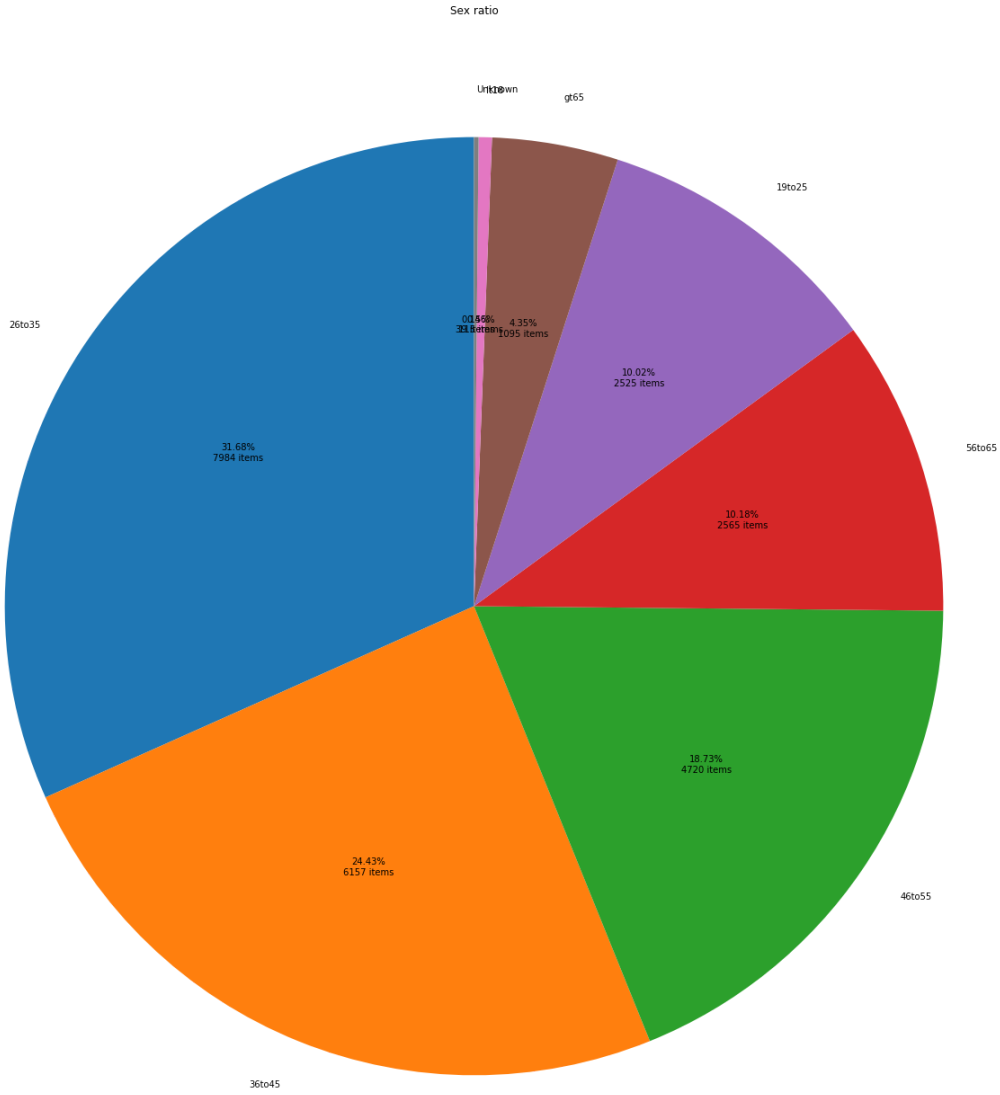
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25200 entries, 0 to 25199
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   step      25200 non-null  object 
 1   customer  25200 non-null  object 
 2   age       25200 non-null  object 
 3   gender    25200 non-null  object 
 4   merchant  25200 non-null  object 
 5   category  25200 non-null  object 
 6   amount    25200 non-null  float64
 7   fraud     25200 non-null  int64  
dtypes: float64(1), int64(1), object(6)
memory usage: 1.5+ MB
```

In [32]:

```
def plot_pie(value, title, label=None, count=True, sort=False, legend=False):
    plt.figure(figsize=(30,24))
    ax = plt.pie(
        value.value_counts(sort=(label is None)),
        startangle=90,
        labels=(None if legend is True else value.value_counts(sort=(label is None)).to_fra
        autopct=(
            lambda p: f'{p:.2f}%\n{p*sum(value.value_counts())/100 :.0f} items' if count is
            else f'{p:.2f}%'
        ),
        pctdistance=0.6,
    )
    if legend:
        plt.legend(labels=label, loc="best", bbox_to_anchor=(1.1, 0., 0.5, 0.5))
    plt.title(title)
    plt.show()
```

In [33]:

```
plot_pie(sample["age"], title="age ratio")   # from this we see that majority of the client
```



In [36]:

```
max(sample['amount'])
```

Out[36]:

8329.96

In [37]:

```
min(sample['amount'])
```

Out[37]:

0.0

In [38]:

```
sum(sample['amount'])
```

Out[38]:

4390155.159999998

In [40]:

```
len(sample['amount'])
```

Out[40]:

25200

In [41]:

```
average=sum(sample['amount'])/len(sample['amount'])
average
```

Out[41]:

174.21250634920628

In [42]:

```
duplicate_row_sample=sample[sample.duplicated()]
duplicate_row_sample
```

Out[42]:

	step	customer	age	gender	merchant	category	amount	fraud

In [46]:

```
print('number of duplicate rows:',duplicate_row_sample.shape)
```

number of duplicate rows: (0, 8)

In [47]:

```
sample
```

Out[47]:

	step	customer	age	gender	merchant	category	amount	fraud
0	2021-07-16	C1326593353	36to45	Male	M1823072687	Transportation	15.78	0
1	2021-08-22	C483912131	26to35	Female	M1198415165	Beauty&Wellness	608.32	1
2	2021-06-03	C1681589600	26to35	Female	M348934600	Transportation	43.76	0
3	2021-07-01	C1251749294	36to45	Female	M1823072687	Transportation	41.57	0
4	2021-03-06	C1705346216	36to45	Male	M348934600	Transportation	42.97	0
...	...	...	...	...	...	...	...	...
25195	2021-08-22	C1480193980	46to55	Female	M1823072687	Transportation	32.07	0
25196	2021-08-14	C207380283	36to45	Female	M480139044	Health	373.27	1
25197	2021-07-18	C183270065	lt18	Female	M1823072687	Transportation	2.50	0
25198	2021-07-23	C149698808	26to35	Female	M1823072687	Transportation	5.38	0
25199	2021-07-16	C684162308	56to65	Female	M840466850	Technology	93.89	1

25200 rows × 8 columns

In [48]:

```
print('customer')
print(list(sample.customer.unique()))
```

```
customer
['C1326593353', 'C483912131', 'C1681589600', 'C1251749294', 'C1705346216',
'C34676686', 'C760368405', 'C1655037147', 'C1789457872', 'C477542874', 'C7
63681944', 'C413455659', 'C1293579037', 'C1453917071', 'C1499701993', 'C36
7052603', 'C126603077', 'C1620825260', 'C449179311', 'C33256704', 'C174039
3527', 'C480961011', 'C1089848865', 'C1978250683', 'C1825950751', 'C911364
935', 'C1059874896', 'C1356412685', 'C140648396', 'C1134994937', 'C1732752
834', 'C1232523772', 'C1735920391', 'C731767226', 'C1499241933', 'C1711218
892', 'C1994178184', 'C1419235351', 'C1642006830', 'C658175205', 'C6620150
24', 'C1491844959', 'C10745331', 'C644590864', 'C2029865097', 'C199844218
3', 'C1703648667', 'C947010068', 'C1156188729', 'C2025967620', 'C191128170
1', 'C1394454342', 'C1729635847', 'C612005121', 'C1188590334', 'C194998468
5', 'C2004941826', 'C808977147', 'C806399525', 'C1794686407', 'C110036122
7', 'C998987490', 'C1255149927', 'C1125717761', 'C506520283', 'C184904634
5', 'C1464489812', 'C1275518867', 'C1803182614', 'C1787537369', 'C3768262
8', 'C1948728559', 'C332229910', 'C706956926', 'C983659750', 'C197798791',
'C1572610482', 'C1348847019', 'C1345265337', 'C422003518', 'C1282114843',
'C1480064437', 'C1559339990', 'C1886871597', 'C1562537029', 'C396866523',
'C475624028', 'C203163727', 'C1082769360', 'C1093842718', 'C190128751', 'C
```

In [49]:

```
print('merchant')
print(list(sample.merchant.unique()))
```

```
merchant
['M1823072687', 'M1198415165', 'M348934600', 'M980657600', 'M480139044', 'M3
697346', 'M85975013', 'M2011752106', 'M547558035', 'M855959430', 'M73219578
2', 'M1888755466', 'M1535107174', 'M349281107', 'M840466850', 'M1353266412',
'M1913465890', 'M151143676', 'M17379832', 'M1649169323', 'M1053599405', 'M21
22776122', 'M45060432', 'M1294758098', 'M1873032707', 'M692898500', 'M857378
720', 'M1600850729', 'M209847108', 'M1741626453', 'M923029380', 'M194609177
8', 'M78078399', 'M2080407379', 'M677738360', 'M50039827', 'M1400236507', 'M
1842530320', 'M1748431652', 'M495352832', 'M1872033263', 'M1313686961', 'M97
925176', 'M1352454843', 'M1416436880', 'M933210764', 'M1788569036', 'M348875
670']
```

In [51]:

```
# adding more features
sample['sold']=sample['amount'].apply(lambda x:x[:4]).astype(float)
sample.head()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-51-0221bf241551> in <module>
      1 # adding more features
----> 2 sample['sold']=sample['amount'].apply(lambda x:x[:4]).astype(float)
      3 sample.head()

~\anaconda3\lib\site-packages\pandas\core\series.py in apply(self, func, con
vert_dtype, args, **kwds)
   3846             else:
   3847                 values = self.astype(object).values
-> 3848                 mapped = lib.map_infer(values, f, convert=convert_dt
ype)
   3849
   3850         if len(mapped) and isinstance(mapped[0], Series):

pandas\_libs\lib.pyx in pandas._libs.lib.map_infer()

<ipython-input-51-0221bf241551> in <lambda>(x)
      1 # adding more features
----> 2 sample['sold']=sample['amount'].apply(lambda x:x[:4]).astype(float)
      3 sample.head()

TypeError: 'float' object is not subscriptable
```

In [7]:

```
a=sample.groupby(['category'=='Transportation'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-7-2c52fda2efbd> in <module>
----> 1 a=sample.groupby(['category'=='Transportation'])

~\anaconda3\lib\site-packages\pandas\core\frame.py in groupby(self, by, axi
s, level, as_index, sort, group_keys, squeeze, observed)
   5808             group_keys=group_keys,
   5809             squeeze=squeeze,
-> 5810             observed=observed,
   5811         )
   5812

~\anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in __init__(sel
f, obj, keys, axis, level, grouper, exclusions, selection, as_index, sort, g
roup_keys, squeeze, observed, mutated)
    407                 sort=sort,
    408                 observed=observed,
--> 409                 mutated=self.mutated,
    410             )
    411

~\anaconda3\lib\site-packages\pandas\core\groupby\grouper.py in get_grouper
(obj, key, axis, level, sort, observed, mutated, validate)
    596                 in_axis, name, level, gpr = False, None, gpr, None
    597             else:
--> 598                 raise KeyError(gpr)
    599         elif isinstance(gpr, Grouper) and gpr.key is not None:
    600             # Add key to exclusions

KeyError: False
```

In [6]:

```
print(a)
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000020D7E659B08>
```

In [ ]: