Project 1 - Credit Fraud

Load in data and create visual to see possible trends

In [35]:
```
#imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn import naive_bayes
from sklearn import utils
```

In [36]:
```
#reading files
df = pd.read_csv("D:/GitHub/data/dsc680/fraudTrain.csv")
df.head()
```

Out[36]:

	Unnamed: 0	trans_date_trans_time	cc_num	merchant	category	amt	fi
0	0	2019-01-01 00:00:18	2703186189652095	fraud_Rippin, Kub and Mann	misc_net	4.97	Jenn
1	1	2019-01-01 00:00:44	630423337322	fraud_Heller, Gutmann and Zieme	grocery_pos	107.23	Stepha
2	2	2019-01-01 00:00:51	38859492057661	fraud_Lind-Buckridge	entertainment	220.11	Edw
3	3	2019-01-01 00:01:16	3534093764340240	fraud_Kutch, Hermiston and Farrell	gas_transport	45.00	Jere
4	4	2019-01-01 00:03:06	375534208663984	fraud_Keeling-Crist	misc_pos	41.96	Ty

In [37]:
```
#check num of na
df.isna().sum().sum()
```

Out[37]: 0

In [38]:
```
#removing index column
df = df.iloc[:,1:]
print(df.head())
print(df.columns)
```

```
  trans_date_trans_time            cc_num                            merchant  \
0   2019-01-01 00:00:18  2703186189652095          fraud_Rippin, Kub and Mann
1   2019-01-01 00:00:44      630423337322     fraud_Heller, Gutmann and Zieme
2   2019-01-01 00:00:51    38859492057661                fraud_Lind-Buckridge
3   2019-01-01 00:01:16  3534093764340240  fraud_Kutch, Hermiston and Farrell
4   2019-01-01 00:03:06   375534208663984                 fraud_Keeling-Crist

        category     amt      first     last gender  \
0       misc_net    4.97   Jennifer    Banks      F
1    grocery_pos  107.23  Stephanie     Gill      F
2  entertainment  220.11     Edward  Sanchez      M
3  gas_transport   45.00     Jeremy    White      M
4       misc_pos   41.96      Tyler   Garcia      M

                         street            city state    zip      lat  \
0                561 Perry Cove  Moravian Falls    NC  28654  36.0788
1  43039 Riley Greens Suite 393          Orient    WA  99160  48.8878
2      594 White Dale Suite 530      Malad City    ID  83252  42.1808
3   9443 Cynthia Court Apt. 038         Boulder    MT  59632  46.2306
4              408 Bradley Rest        Doe Hill    VA  24433  38.4207

       long  city_pop                                job         dob  \
0  -81.1781      3495          Psychologist, counselling  1988-03-09
1 -118.2105       149  Special educational needs teacher  1978-06-21
2 -112.2620      4154        Nature conservation officer  1962-01-19
3 -112.1138      1939                    Patent attorney  1967-01-12
4  -79.4629        99      Dance movement psychotherapist  1986-03-28

                          trans_num   unix_time  merch_lat  merch_long  \
0  0b242abb623afc578575680df30655b9  1325376018  36.011293  -82.048315
1  1f76529f8574734946361c461b024d99  1325376044  49.159047 -118.186462
2  a1a22d70485983eac12b5b88dad1cf95  1325376051  43.150704 -112.154481
3  6b849c168bdad6f867558c3793159a81  1325376076  47.034331 -112.561071
4  a41d7549acf90789359a9aa5346dcb46  1325376186  38.674999  -78.632459

   is_fraud
0         0
1         0
2         0
3         0
4         0
Index(['trans_date_trans_time', 'cc_num', 'merchant', 'category', 'amt',
       'first', 'last', 'gender', 'street', 'city', 'state', 'zip', 'lat',
       'long', 'city_pop', 'job', 'dob', 'trans_num', 'unix_time', 'merch_lat',
       'merch_long', 'is_fraud'],
      dtype='object')
```

In [39]:
```
#check unique credit card number, fraud attacks tend to hit one credit card alot
#print(df['cc_num'].value_counts())
#looking at number of fraud
print("Count of fraud vs non fraud charges\n")
print(df['is_fraud'].value_counts())
#looking at number of fraud charges and unique cards
print("\nNumber of charges per card that has fraud:\n")
print(df[df['is_fraud'] == 1]['cc_num'].value_counts())
#looking at number unique cards that have fraud charges
print("\nNumber of unique cards to have fraud charges:\n")
print(len(df[df['is_fraud'] == 1]['cc_num'].unique()))
```

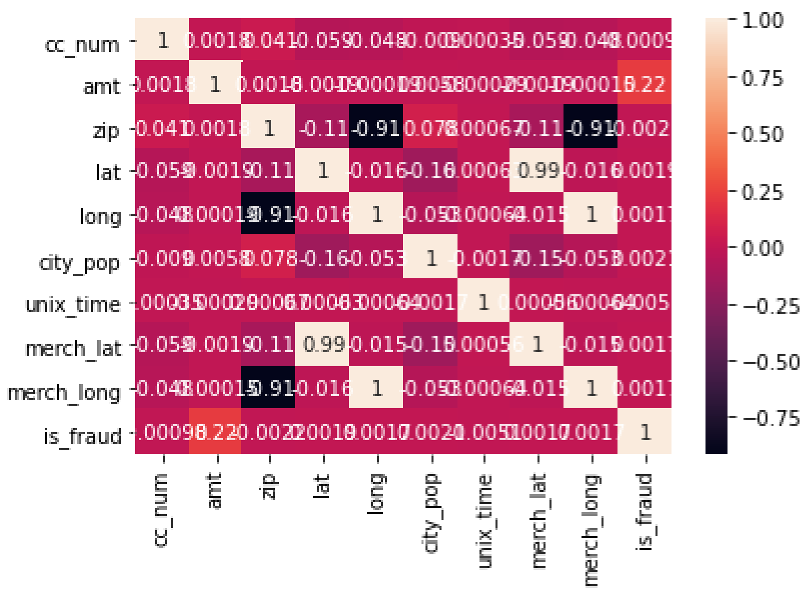
```
Count of fraud vs non fraud charges

0    1289169
1       7506
Name: is_fraud, dtype: int64

Number of charges per card that has fraud:

3520550088202337       19
4593569795412          19
4260128500325          18
4400011257587661852    16
4629451965224809       16
                       ..
4503101193493052864     2
4809701904914           2
4005676619255478        2
6011109736646996        2
4089096483689733451     2
Name: cc_num, Length: 762, dtype: int64

Number of unique cards to have fraud charges:

762
```

In [40]:
```
#Looking at correlation of values
df.corr()
```

Out[40]:

	cc_num	amt	zip	lat	long	city_pop	unix_time	merch_lat
cc_num	1.000000	0.001769	0.041459	-0.059271	-0.048278	-0.008991	0.000354	-0.058942
amt	0.001769	1.000000	0.001843	-0.001926	-0.000187	0.005818	-0.000293	-0.001873
zip	0.041459	0.001843	1.000000	-0.114290	-0.909732	0.078467	0.000670	-0.113561
lat	-0.059271	-0.001926	-0.114290	1.000000	-0.015533	-0.155730	0.000632	0.993592
long	-0.048278	-0.000187	-0.909732	-0.015533	1.000000	-0.052715	-0.000642	-0.015452
city_pop	-0.008991	0.005818	0.078467	-0.155730	-0.052715	1.000000	-0.001714	-0.154781
unix_time	0.000354	-0.000293	0.000670	0.000632	-0.000642	-0.001714	1.000000	0.000561
merch_lat	-0.058942	-0.001873	-0.113561	0.993592	-0.015452	-0.154781	0.000561	1.000000
merch_long	-0.048252	-0.000151	-0.908924	-0.015509	0.999120	-0.052687	-0.000635	-0.015431
is_fraud	-0.000981	0.219404	-0.002162	0.001894	0.001721	0.002136	-0.005078	0.001741

In [41]:
```
sns.heatmap(df.corr(),annot=True)
plt.show()
```



In [42]:
```
df.apply(lambda x: x.factorize()[0]).corr()
```

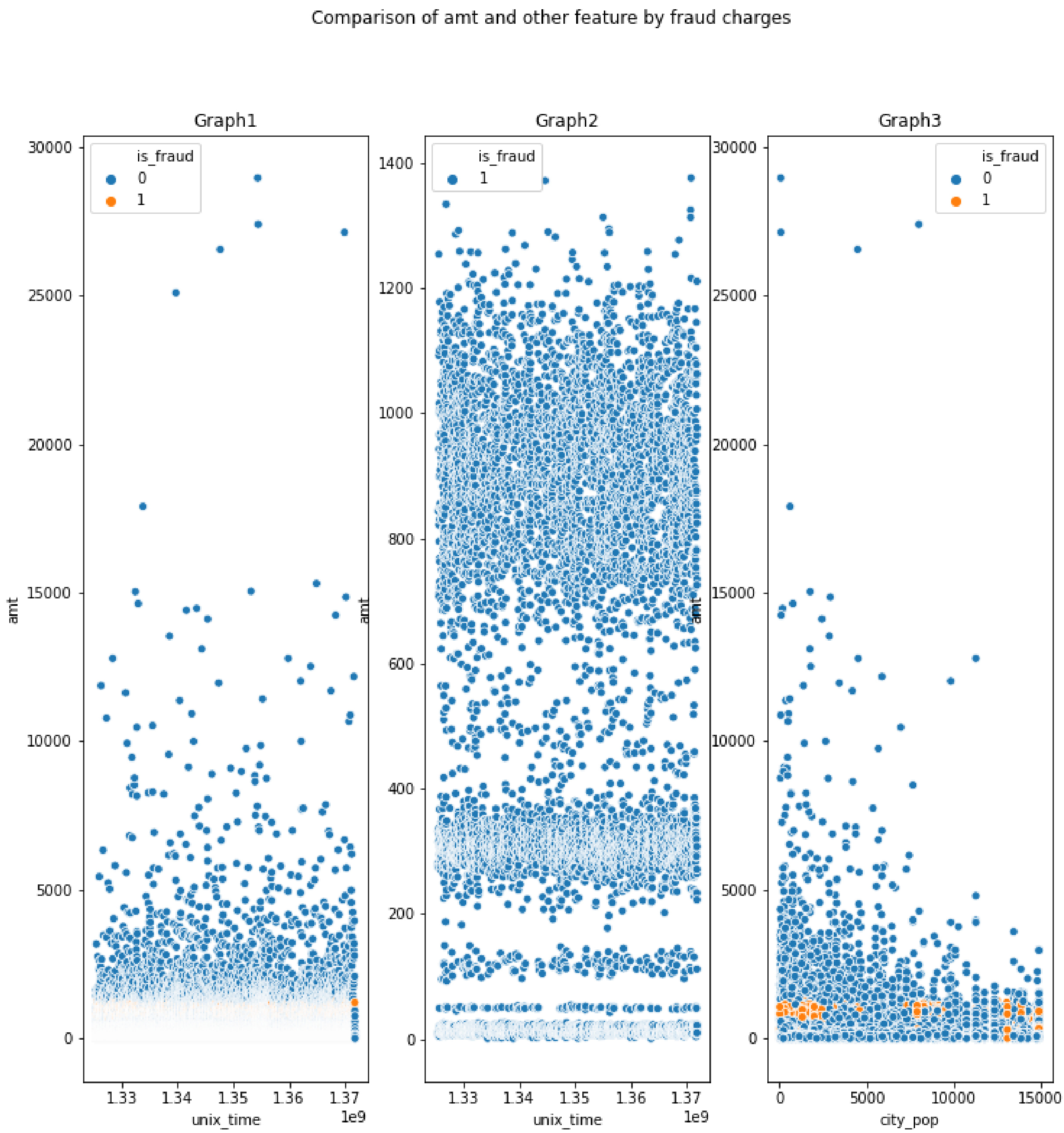
Out[42]:

	trans_date_trans_time	cc_num	merchant	category	amt	first
trans_date_trans_time	1.000000	0.002475	0.001556	-0.000063	0.050232	0.000469
cc_num	0.002475	1.000000	-0.000405	-0.002461	0.003603	0.404395
merchant	0.001556	-0.000405	1.000000	0.600791	-0.053391	-0.002894
category	-0.000063	-0.002461	0.600791	1.000000	-0.077282	-0.002047
amt	0.050232	0.003603	-0.053391	-0.077282	1.000000	-0.007050
first	0.000469	0.404395	-0.002894	-0.002047	-0.007050	1.000000
last	0.001677	0.554262	0.000016	-0.000060	-0.000441	0.286354
gender	-0.000946	0.053981	0.003712	-0.003711	-0.019953	-0.104190
street	0.002475	1.000000	-0.000405	-0.002461	0.003603	0.404395
city	0.002467	0.929132	-0.000895	-0.003157	0.002221	0.389153
state	0.001220	0.086740	-0.001758	-0.001749	-0.010552	0.119001
zip	0.002424	0.985516	-0.000573	-0.002507	0.004085	0.404675
lat	0.002406	0.984086	-0.000631	-0.002551	0.004090	0.404238
long	0.002333	0.982925	-0.000590	-0.002599	0.003665	0.405104
city_pop	0.002414	0.923754	-0.000255	-0.002463	0.002295	0.400056
job	0.001082	0.588202	-0.000163	-0.000254	0.000960	0.294307
dob	0.002574	0.984916	-0.000724	-0.002942	0.002177	0.400812
trans_num	0.999999	0.002474	0.001558	-0.000061	0.050207	0.000469
unix_time	1.000000	0.002475	0.001556	-0.000063	0.050232	0.000469
merch_lat	0.962475	0.003736	0.001378	-0.000071	0.049254	0.000299
merch_long	0.983882	0.002049	0.001323	-0.000097	0.049968	0.000137
is_fraud	-0.004777	0.029358	-0.034868	-0.039249	0.181432	0.009605

In [43]:
```
pd.set_option('display.max_columns', 25)
```

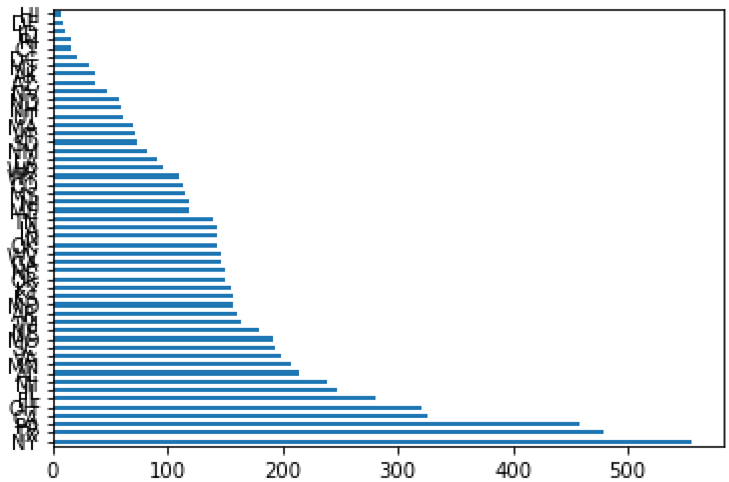
In [44]:
```
fig, axes = plt.subplots(1,3,figsize=(12,12))
fig.suptitle("Comparison of amt and other feature by fraud charges")
sns.scatterplot(ax = axes[0], data=df, x='unix_time', y='amt', hue='is_fraud')
axes[0].set_title("Graph1")
sns.scatterplot(ax = axes[1], data=df[df['is_fraud'] == 1], x='unix_time', y='amt
axes[1].set_title("Graph2")
sns.scatterplot(ax = axes[2], data=df[df['city_pop'] < 15000], x='city_pop', y='a
axes[2].set_title("Graph3")
```

Out[44]:
```
Text(0.5, 1.0, 'Graph3')
```



In [45]:
```
state_count = pd.value_counts(df[df['is_fraud'] == 1]['state'].values)
state_count.plot.barh()
```

Out[45]: <matplotlib.axes._subplots.AxesSubplot at 0x1caef37dee0>



In [46]:
```
#label encoding to try with all sig values
df["merchant"] = df['merchant'].astype('category')
df["category"] = df["category"].astype('category')
df["street"] = df["street"].astype('category')
df["merch"] = df['merchant'].cat.codes
df["cat"] = df["category"].cat.codes
df["st"] = df["street"].cat.codes
df.head()
```

Out[46]:

	trans_date_trans_time	cc_num	merchant	category	amt	first	las
0	2019-01-01 00:00:18	2703186189652095	fraud_Rippin, Kub and Mann	misc_net	4.97	Jennifer	Bank
1	2019-01-01 00:00:44	630423337322	fraud_Heller, Gutmann and Zieme	grocery_pos	107.23	Stephanie	Gi
2	2019-01-01 00:00:51	38859492057661	fraud_Lind-Buckridge	entertainment	220.11	Edward	Sanche
3	2019-01-01 00:01:16	3534093764340240	fraud_Kutch, Hermiston and Farrell	gas_transport	45.00	Jeremy	Whit
4	2019-01-01 00:03:06	375534208663984	fraud_Keeling-Crist	misc_pos	41.96	Tyler	Garci

In [47]:
```
df.corr()
```

Out[47]:

	cc_num	amt	zip	lat	long	city_pop	unix_time	merch_lat
cc_num	1.000000	0.001769	0.041459	-0.059271	-0.048278	-0.008991	0.000354	-0.058942
amt	0.001769	1.000000	0.001843	-0.001926	-0.000187	0.005818	-0.000293	-0.001873
zip	0.041459	0.001843	1.000000	-0.114290	-0.909732	0.078467	0.000670	-0.113561
lat	-0.059271	-0.001926	-0.114290	1.000000	-0.015533	-0.155730	0.000632	0.993592
long	-0.048278	-0.000187	-0.909732	-0.015533	1.000000	-0.052715	-0.000642	-0.015452
city_pop	-0.008991	0.005818	0.078467	-0.155730	-0.052715	1.000000	-0.001714	-0.154781
unix_time	0.000354	-0.000293	0.000670	0.000632	-0.000642	-0.001714	1.000000	0.000561
merch_lat	-0.058942	-0.001873	-0.113561	0.993592	-0.015452	-0.154781	0.000561	1.000000
merch_long	-0.048252	-0.000151	-0.908924	-0.015509	0.999120	-0.052687	-0.000635	-0.015431
is_fraud	-0.000981	0.219404	-0.002162	0.001894	0.001721	0.002136	-0.005078	0.001741
merch	0.000055	-0.002633	0.001113	-0.002266	-0.000697	0.001911	-0.000999	-0.002263
cat	0.001230	0.030867	0.002371	-0.008660	-0.000767	0.009386	0.000182	-0.008519
st	0.046509	0.001346	-0.053860	-0.012651	0.071328	-0.012530	-0.001089	-0.012517

In [48]:
```
#split data so that there are equal part fraud and non fraud
fraud = df[df["is_fraud"] == 1]
nf = df[df["is_fraud"] == 0]

testF = fraud.sample(frac= 0.3)

trainF = fraud.drop(testF.index)

testN = nf.sample(frac= 0.3)

trainN = nf.drop(testN.index)

train = trainF.append(trainN)
test = testF.append(testN)

print(train.head())
print(test.head())
```

```
     trans_date_trans_time            cc_num                     merchant  \
2449   2019-01-02 01:06:37     4613314721966     fraud_Rutherford-Mertz
2546   2019-01-02 03:38:03     4613314721966     fraud_Erdman-Kertzmann
2553   2019-01-02 03:55:47   340187018810220         fraud_Koepp-Parker
3527   2019-01-02 23:52:08     4613314721966       fraud_Ruecker Group
3580   2019-01-03 01:05:27   340187018810220   fraud_Conroy-Cruickshank

           category     amt  first     last gender                     street  \
2449    grocery_pos  281.06  Jason   Murphy      M  542 Steve Curve Suite 011
2546  gas_transport    7.03  Jason   Murphy      M  542 Steve Curve Suite 011
2553    grocery_pos  275.73  Misty     Hart      F  27954 Hall Mill Suite 575
3527       misc_net  843.91  Jason   Murphy      M  542 Steve Curve Suite 011
3580  gas_transport   10.76  Misty     Hart      F  27954 Hall Mill Suite 575

               city state    zip      lat     long  city_pop  \
2449  Collettsville    NC  28611  35.9946 -81.7266       885
2546  Collettsville    NC  28611  35.9946 -81.7266       885
2553    San Antonio    TX  78208  29.4400 -98.4590   1595797
3527  Collettsville    NC  28611  35.9946 -81.7266       885
3580    San Antonio    TX  78208  29.4400 -98.4590   1595797

                            job         dob                         trans_num  \
2449             Soil scientist  1988-09-15  e8a81877ae9a0a7f883e15cb39dc4022
2546             Soil scientist  1988-09-15  397894a5c4c02e3c61c784001f0f14e4
2553  Horticultural consultant  1960-10-28  7863235a750d73a244c07f1fb7f0185a
3527             Soil scientist  1988-09-15  2f7d497f607396ab669c14c2abe3886f
3580  Horticultural consultant  1960-10-28  0a2f8002e55a3565c5c88d8cf039fed8

       unix_time  merch_lat  merch_long  is_fraud  merch  cat   st
2449  1325466397  36.430124  -81.179483         1    543    4  544
2546  1325475483  35.909292  -82.091010         1    162    2  544
2553  1325476547  29.786426  -98.683410         1    328    4  280
3527  1325548328  35.985612  -81.383306         1    535    8  544
3580  1325552727  28.856712  -97.794207         1    103    2  280
        trans_date_trans_time              cc_num  \
396353   2019-06-30 02:47:30  4265776278887457
760769   2019-11-22 00:20:13    30044330818990
```

```
252784   2019-05-06 03:06:47   180056173248083
890453   2019-12-23 23:43:07  6011581063717667
601843   2019-09-13 00:09:43   374238209524200

                                 merchant      category     amt      first  \
396353              fraud_Rutherford-Mertz   grocery_pos  333.34  Christine
760769                    fraud_Conroy Ltd  shopping_pos  717.14    Allison
252784  fraud_Schultz, Simonis and Little   grocery_pos  277.17      Larry
890453                   fraud_Pouros-Haag  shopping_pos  846.92      Jerry
601843  fraud_Schultz, Simonis and Little   grocery_pos  321.76     Daniel

            last gender                       street        city state    zip
\
396353      Best      F             68248 Deanna Land       Enola    AR  72047
760769     Ayala      F    87665 Karen Mill Apt. 586  Fort Myers    FL  33967
252784    Warner      M    145 Jeffrey Key Suite 668    Lakeview    MI  48850
890453   Perkins      M  3867 Susan Corners Apt. 883    Brashear    MO  63533
601843  Martinez      M            8510 Acevedo Burgs        Kent    OR  97033

            lat      long  city_pop                                  job  \
396353  35.2087  -92.2123       969                   Physicist, medical
760769  26.4722  -81.8122    224256                            Paramedic
252784  43.4269  -85.2924      4474  Emergency planning/management officer
890453  40.1959  -92.4333       805                 Private music teacher
601843  45.0838 -120.6649        60               Museum education officer

               dob                         trans_num   unix_time  merch_lat  \
396353  1954-01-05  fbbf055ee4093d1e13263f44d16bd742  1341024450  34.212793
760769  1985-08-29  57fe73500386c72c42509d3b3f22af94  1353543613  25.839930
252784  1976-01-15  315d39ecf211723e026ab3b14cad4642  1336273607  43.281135
890453  1970-06-27  5c2acede6d48fbd09d984e4f15ee434f  1356306187  40.642867
601843  1942-04-03  4ee4ce1d64ac8f9efb53ec9118e8b579  1347494983  45.748081

        merch_long  is_fraud  merch  cat   st
396353  -91.709136         1    543    4  681
760769  -80.839343         1    101   12  872
252784  -85.079532         1    570    4  139
890453  -91.793663         1    484   12  404
601843 -119.749367         1    570    4  844
```

In [49]:

```
x_train = train[["cc_num","amt","zip","long","lat"]]
y_train = train["is_fraud"]

x_test = test[["cc_num","amt","zip","long","lat"]]
y_test = test["is_fraud"]
```

In [50]:
```
#creating function to analyze classifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
# Print out common error metrics for the binary classifications.
def print_multiclass_classif_error_report(y_test, preds):
    print('Accuracy: ' + str(accuracy_score(y_test, preds)))
    print('Avg. F1 (Micro): ' + str(f1_score(y_test, preds, average='micro')))
    print('Avg. F1 (Macro): ' + str(f1_score(y_test, preds, average='macro')))
    print('Avg. F1 (Weighted): ' + str(f1_score(y_test, preds, average='weighted'
    print(classification_report(y_test, preds))
    print("Confusion Matrix:\n" + str(confusion_matrix(y_test, preds)))
```

In [51]:
```
#naive bayes model
gnb_mod = naive_bayes.GaussianNB()
gnb_mod.fit(x_train,y_train)

pred = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, pred)
```

```
Accuracy: 0.9942108415616332
Avg. F1 (Micro): 0.9942108415616332
Avg. F1 (Macro): 0.4985485089345334
Avg. F1 (Weighted): 0.9913246652541997

C:\Users\Matt Kline\anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1221: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` paramet
er to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

              precision    recall  f1-score   support

           0       0.99      1.00      1.00    386751
           1       0.00      0.00      0.00      2252

    accuracy                           0.99    389003
   macro avg       0.50      0.50      0.50    389003
weighted avg       0.99      0.99      0.99    389003

Confusion Matrix:
[[386751      0]
 [  2252      0]]
```

In [52]:
```
#going to run same code as above but with data that is equal part fraud and no fr
fraud = df[df["is_fraud"] == 1]
no = df[df["is_fraud"] == 0]

no = no.sample(n=len(fraud))

print(len(fraud))
print(len(no))

df2 = fraud
df2 = df2.append(no)
df2 = utils.shuffle(df2)
df2.head()
```

```
7506
7506
```

Out[52]:

	trans_date_trans_time	cc_num	merchant	category	amt	fi
27670	2019-01-16 23:32:14	6011366578560244	fraud_Romaguera, Cruickshank and Greenholt	shopping_net	1125.31	Ad
2552	2019-01-02 03:55:32	4878364946692291	fraud_Schmitt Ltd	misc_net	5.37	T
847913	2019-12-15 05:07:11	30427035050508	fraud_Rowe, Batz and Goodwin	grocery_pos	52.20	Jc
506403	2019-08-07 22:50:50	6011388901471808	fraud_Boyer PLC	shopping_net	1120.05	Jacquel
1027652	2020-03-02 01:07:38	345389171551808	fraud_DuBuque LLC	grocery_pos	327.98	Jus

In [53]:
```
x = df2[["cc_num","amt","zip","long","lat"]]
y = df2["is_fraud"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, randor

gnb_mod = naive_bayes.GaussianNB()

gnb_mod.fit(x_train, y_train)

preds = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, preds)
```

```
Accuracy: 0.5037744227353463
Avg. F1 (Micro): 0.5037744227353463
Avg. F1 (Macro): 0.40345632938295983
Avg. F1 (Weighted): 0.40247867606387866
              precision    recall  f1-score   support

           0       0.53      0.09      0.16      2261
           1       0.50      0.92      0.65      2243

    accuracy                           0.50      4504
   macro avg       0.52      0.51      0.40      4504
weighted avg       0.52      0.50      0.40      4504

Confusion Matrix:
[[ 211 2050]
 [ 185 2058]]
```

In [54]:
```
#label encoding to try with all sig values
df2["merchant"] = df2['merchant'].astype('category')
df2["category"] = df2["category"].astype('category')
df2["street"] = df2["street"].astype('category')
df2["merch"] = df2['merchant'].cat.codes
df2["cat"] = df2["category"].cat.codes
df2["st"] = df2["street"].cat.codes
df2.head()
```

Out[54]:

	trans_date_trans_time	cc_num	merchant	category	amt	fi
27670	2019-01-16 23:32:14	6011366578560244	fraud_Romaguera, Cruickshank and Greenholt	shopping_net	1125.31	Ad
2552	2019-01-02 03:55:32	4878364946692291	fraud_Schmitt Ltd	misc_net	5.37	T
847913	2019-12-15 05:07:11	30427035050508	fraud_Rowe, Batz and Goodwin	grocery_pos	52.20	Jo
506403	2019-08-07 22:50:50	6011388901471808	fraud_Boyer PLC	shopping_net	1120.05	Jacquel
1027652	2020-03-02 01:07:38	345389171551808	fraud_DuBuque LLC	grocery_pos	327.98	Jus

In [55]:
```
#try with included columns
x = df2[["cc_num","amt","zip","long","lat","merch","cat","st"]]
y = df2["is_fraud"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, randon

gnb_mod = naive_bayes.GaussianNB()

gnb_mod.fit(x_train, y_train)

preds = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, preds)
```

```
Accuracy: 0.5037744227353463
Avg. F1 (Micro): 0.5037744227353463
Avg. F1 (Macro): 0.40345632938295983
Avg. F1 (Weighted): 0.40247867606387866
              precision    recall  f1-score   support

           0       0.53      0.09      0.16      2261
           1       0.50      0.92      0.65      2243

    accuracy                           0.50      4504
   macro avg       0.52      0.51      0.40      4504
weighted avg       0.52      0.50      0.40      4504

Confusion Matrix:
[[ 211 2050]
 [ 185 2058]]
```

In [56]:
```
df2.corr()
```

Out[56]:

	cc_num	amt	zip	lat	long	city_pop	unix_time	merch_lat
cc_num	1.000000	0.014889	0.040887	-0.041670	-0.040556	-0.009842	0.006858	-0.043060
amt	0.014889	1.000000	-0.017854	0.009435	0.014234	0.018743	-0.004959	0.008849
zip	0.040887	-0.017854	1.000000	-0.087938	-0.912835	0.103120	0.002795	-0.087147
lat	-0.041670	0.009435	-0.087938	1.000000	-0.060626	-0.170461	-0.018914	0.993639
long	-0.040556	0.014234	-0.912835	-0.060626	1.000000	-0.074231	0.002106	-0.060983
city_pop	-0.009842	0.018743	0.103120	-0.170461	-0.074231	1.000000	-0.003356	-0.170098
unix_time	0.006858	-0.004959	0.002795	-0.018914	0.002106	-0.003356	1.000000	-0.019259
merch_lat	-0.043060	0.008849	-0.087147	0.993639	-0.060983	-0.170098	-0.019259	1.000000
merch_long	-0.040636	0.014435	-0.911881	-0.060236	0.999142	-0.074012	0.002312	-0.060577
is_fraud	-0.005059	0.601883	-0.014545	0.013482	0.011141	0.010562	-0.025149	0.012566
merch	0.001217	-0.000757	-0.004141	0.005094	0.006040	-0.001078	0.009401	0.006220
cat	0.004730	0.432095	-0.003625	-0.008171	0.003334	0.010077	0.003533	-0.008341
st	0.001014	0.004255	-0.026559	0.012806	0.041241	-0.011907	-0.000660	0.013009

In [57]:
```
x = df2[["cc_num","amt","zip","long","lat","merch_long","merch_lat","cat"]]
y = df2["is_fraud"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, randor

gnb_mod = naive_bayes.GaussianNB()

gnb_mod.fit(x_train, y_train)

preds = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, preds)
```

```
Accuracy: 0.5037744227353463
Avg. F1 (Micro): 0.5037744227353463
Avg. F1 (Macro): 0.40345632938295983
Avg. F1 (Weighted): 0.40247867606387866
              precision    recall  f1-score   support

           0       0.53      0.09      0.16      2261
           1       0.50      0.92      0.65      2243

    accuracy                           0.50      4504
   macro avg       0.52      0.51      0.40      4504
weighted avg       0.52      0.50      0.40      4504

Confusion Matrix:
[[ 211 2050]
 [ 185 2058]]
```

In [58]:
```
x = df2[["amt","cat"]]
y = df2["is_fraud"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, randon

gnb_mod = naive_bayes.GaussianNB()

gnb_mod.fit(x_train, y_train)

preds = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, preds)
```

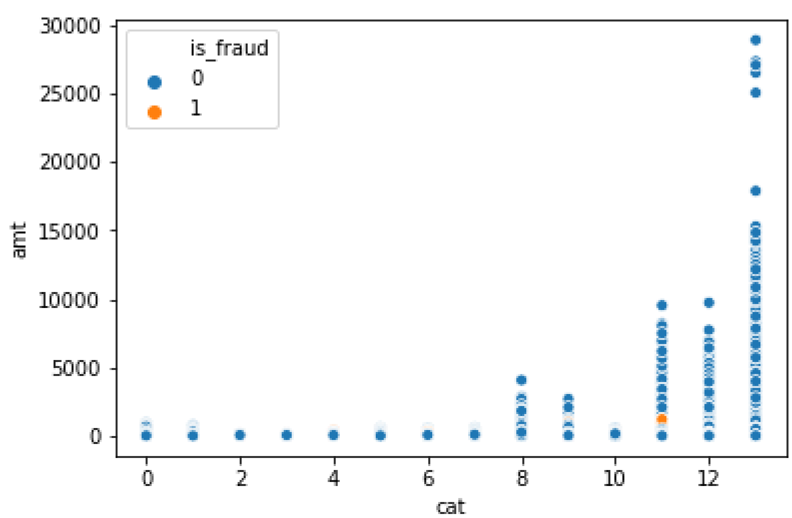
```
Accuracy: 0.7495559502664298
Avg. F1 (Micro): 0.7495559502664298
Avg. F1 (Macro): 0.7354508463053497
Avg. F1 (Weighted): 0.7356949731046761
              precision    recall  f1-score   support

           0       0.67      0.98      0.80      2261
           1       0.96      0.52      0.67      2243

    accuracy                           0.75      4504
   macro avg       0.81      0.75      0.74      4504
weighted avg       0.81      0.75      0.74      4504

Confusion Matrix:
[[2208   53]
 [1075 1168]]
```

In [59]:
```
#trying original size model with best smaller modal features
x_train = train[["amt","cat"]]
y_train = train["is_fraud"]

x_test = test[["amt","cat"]]
y_test = test["is_fraud"]
#naive bayes model
gnb_mod = naive_bayes.GaussianNB()
gnb_mod.fit(x_train,y_train)

pred = gnb_mod.predict(x_test)
print_multiclass_classif_error_report(y_test, pred)
```

```
Accuracy: 0.9909460852486999
Avg. F1 (Micro): 0.9909460852486999
Avg. F1 (Macro): 0.6826933068109711
Avg. F1 (Weighted): 0.991819197316844
              precision    recall  f1-score   support

           0       1.00      0.99      1.00    386751
           1       0.31      0.46      0.37      2252

    accuracy                           0.99    389003
   macro avg       0.65      0.73      0.68    389003
weighted avg       0.99      0.99      0.99    389003

Confusion Matrix:
[[384447   2304]
 [  1218   1034]]
```

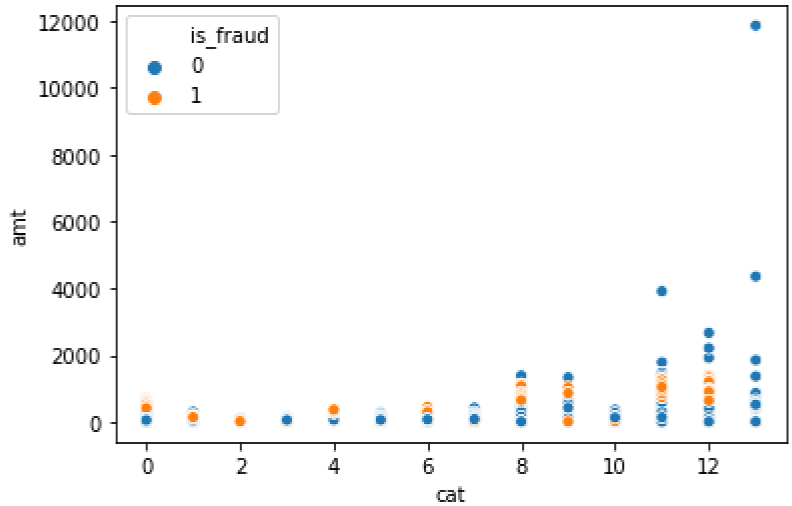
In [60]:
```
sns.scatterplot(data=df, x='cat', y='amt', hue='is_fraud')
```

Out[60]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1caef33caf0>
```



In [61]:
```
sns.scatterplot(data=df2, x='cat', y='amt', hue='is_fraud')
```

Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x1caef31f6a0>



In [ ]: