In [1]:

```
import pandas as pd
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline
```

In [2]:

```
%time
fa_dir = '/Users/stevecoggeshall/Documents/Teaching/Fraud Analytics/2018 USC fraud c
mydata = pd.read_excel(fa_dir + '/data/card transactions/card transactions.xlsx')
```

```
CPU times: user 3 µs, sys: 1 µs, total: 4 µs
Wall time: 7.87 µs
```

In [3]:

```
mydata.dtypes
```

Out[3]:

```
Recordnum                       int64
Cardnum                         int64
Date                   datetime64[ns]
Merchantnum                    object
Merch Description              object
Merchant State                 object
Merchant Zip                  float64
Transtype                      object
Amount                        float64
Fraud                           int64
dtype: object
```

In [4]:

```
mydata.head(10)
```

Out[4]:

	Recordnum	Cardnum	Date	Merchantnum	Merch Description	Merchant State	Merchant Zip
0	1	5142190439	2010-01-01	5509006296254	FEDEX SHP 12/23/09 AB#	TN	38118.0
1	2	5142183973	2010-01-01	61003026333	SERVICE MERCHANDISE #81	MA	1803.0
2	3	5142131721	2010-01-01	4503082993600	OFFICE DEPOT #191	MD	20706.0
3	4	5142148452	2010-01-01	5509006296254	FEDEX SHP 12/28/09 AB#	TN	38118.0
4	5	5142190439	2010-01-01	5509006296254	FEDEX SHP 12/23/09 AB#	TN	38118.0
5	6	5142149874	2010-01-01	5509006296254	FEDEX SHP 12/22/09 AB#	TN	38118.0
6	7	5142189277	2010-01-01	5509006296254	FEDEX SHP 12/28/09 AB#	TN	38118.0
7	8	5142191182	2010-01-01	6098208200062	MIAMI COMPUTER SUPPLY	OH	45429.0
8	9	5142258629	2010-01-01	602608969534	FISHER SCI ATL	GA	30091.0
9	10	5142190439	2010-01-01	5509006296254	FEDEX SHP 12/23/09 AB#	TN	38118.0

In [5]:

```
def mem_usage(pandas_obj):
    if isinstance(pandas_obj,pd.DataFrame):
        usage_b = pandas_obj.memory_usage(deep=True).sum()
    else: # we assume if not a df it's a series
        usage_b = pandas_obj.memory_usage(deep=True)
    usage_mb = usage_b / 1024 ** 2 # convert bytes to megabytes
    return "{:03.2f} MB".format(usage_mb)
```

In [6]:

```
print(mem_usage(mydata))
```

```
29.18 MB
```

In [7]:

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

```
/Users/stevecoggeshall/anaconda3/lib/python3.5/site-packages/numpy/lib
/function_base.py:4291: RuntimeWarning: Invalid value encountered in p
ercentile
  interpolation=interpolation)
```

Out[7]:

	Recordnum	Cardnum	Date	Merchantnum	Merch Description	Merchant State	Me
count	96708.000000	9.670800e+04	96708	93333	96708	95513	920
unique	NaN	NaN	365	13090	13125	227	Nal
top	NaN	NaN	2010-02-28 00:00:00	930090121224	GSA-FSS-ADV	TN	Nal
freq	NaN	NaN	684	9310	1688	11990	Nal
first	NaN	NaN	2010-01-01 00:00:00	NaN	NaN	NaN	Nal
last	NaN	NaN	2010-12-31 00:00:00	NaN	NaN	NaN	Nal
mean	48354.500000	5.142201e+09	NaN	NaN	NaN	NaN	447
std	27917.339254	5.391327e+04	NaN	NaN	NaN	NaN	283
min	1.000000	5.142110e+09	NaN	NaN	NaN	NaN	1.0
25%	24177.750000	5.142152e+09	NaN	NaN	NaN	NaN	Nal
50%	48354.500000	5.142196e+09	NaN	NaN	NaN	NaN	Nal
75%	72531.250000	5.142246e+09	NaN	NaN	NaN	NaN	Nal
max	96708.000000	5.142311e+09	NaN	NaN	NaN	NaN	999

In [8]:

```
numrecords = len(mydata)
print(numrecords)
```

```
96708
```

In [9]:

```
mydata.count() * 100 /numrecords
```

Out[9]:

```
Recordnum            100.000000
Cardnum              100.000000
Date                 100.000000
Merchantnum           96.510113
Merch Description    100.000000
Merchant State        98.764321
Merchant Zip          95.185507
Transtype            100.000000
Amount               100.000000
Fraud                100.000000
dtype: float64
```

In [10]:

```
len(mydata['Recordnum'].unique())
```

Out[10]:

```
96708
```

In [11]:

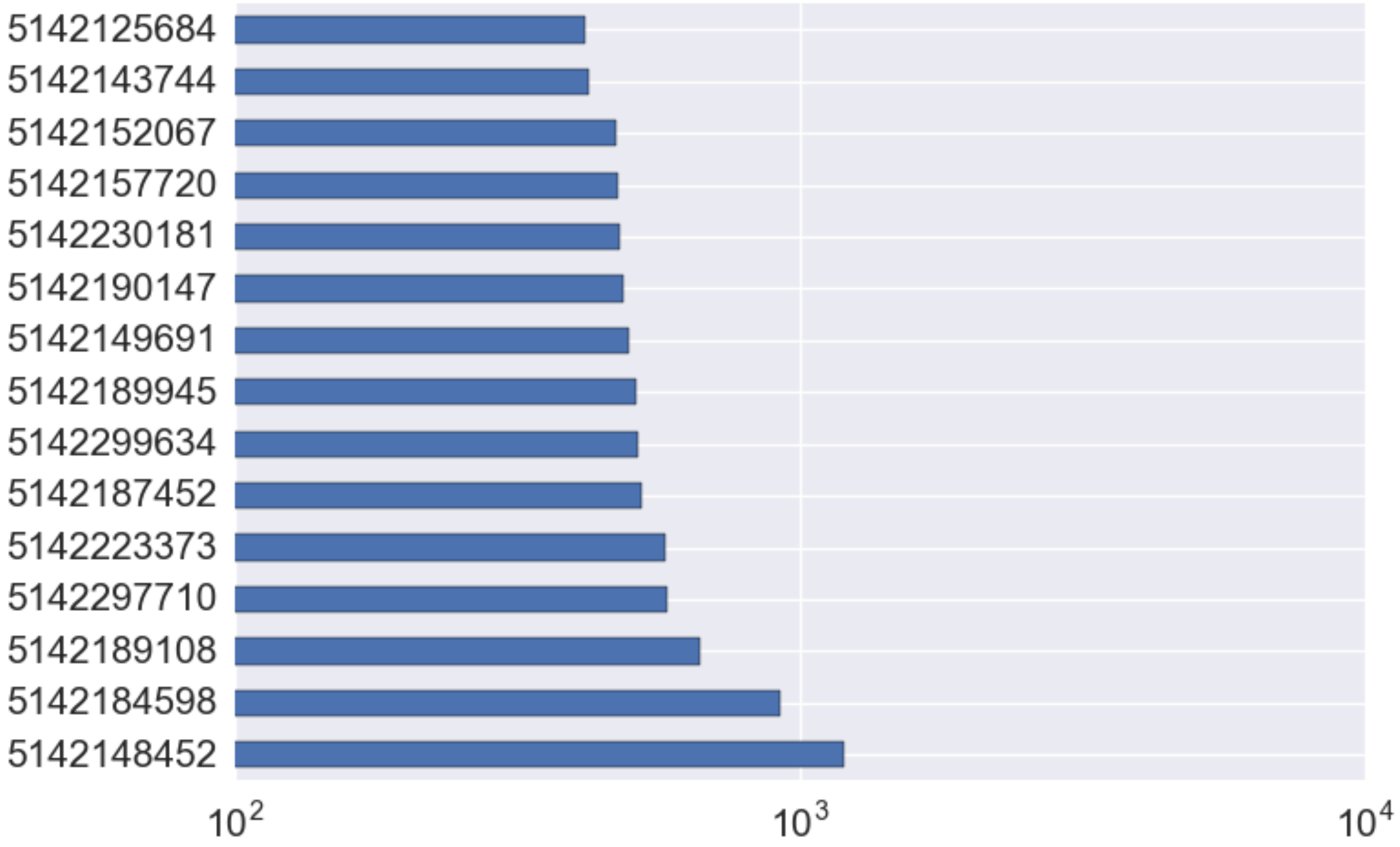
```
len(mydata['Cardnum'].unique())
```

Out[11]:

```
1644
```

In [12]:

```
sns.set(font_scale=1.5)
mydata['Cardnum'].value_counts().head(15).plot(kind = 'barh')
plt.xscale('log')
```



In [13]:

```
len(mydata['Date'].unique())
```

Out[13]:

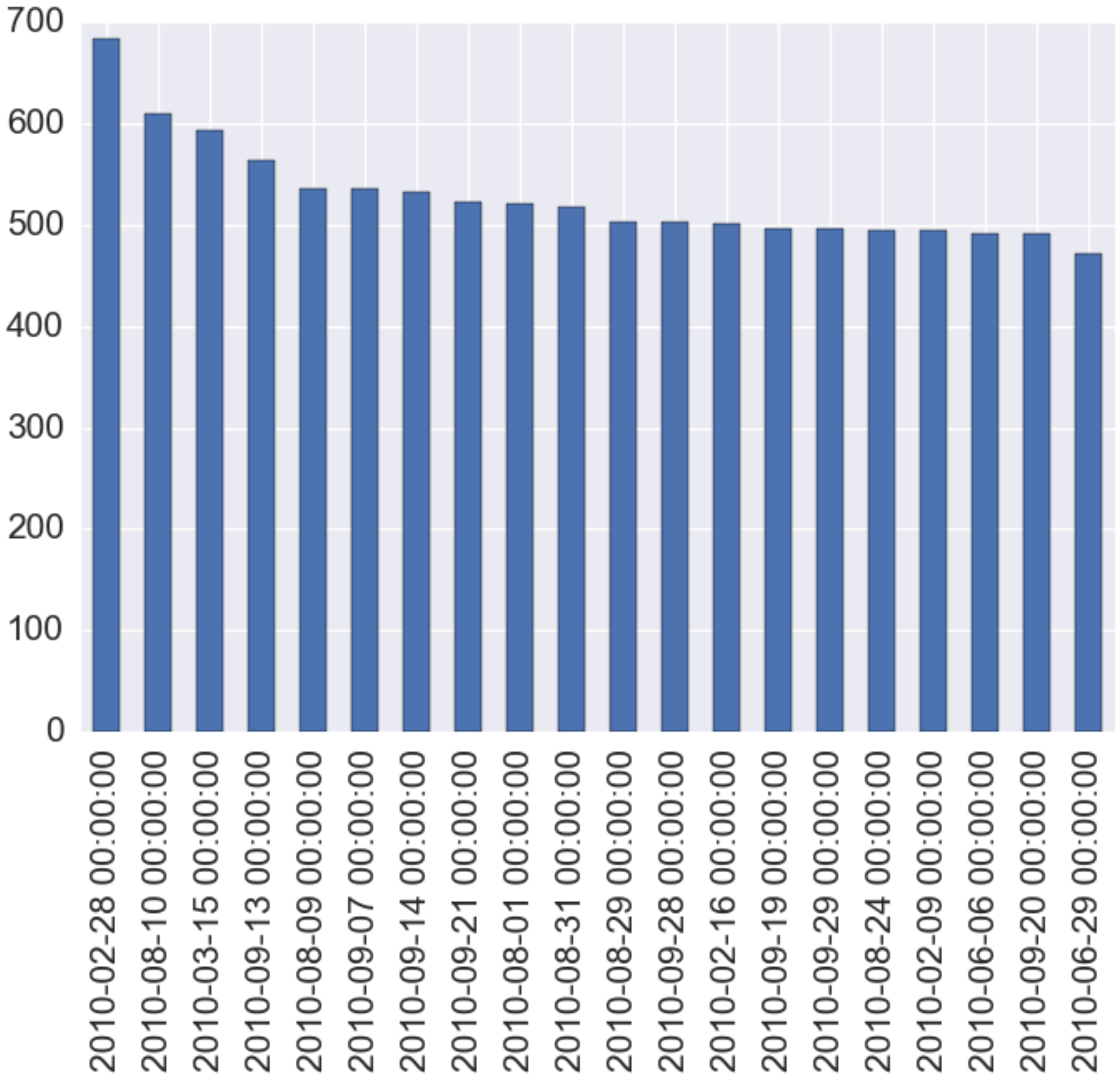
```
365
```

In [14]:

```
mydata['Date'].value_counts().head(20).plot(kind = 'bar')
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x119478160>
```

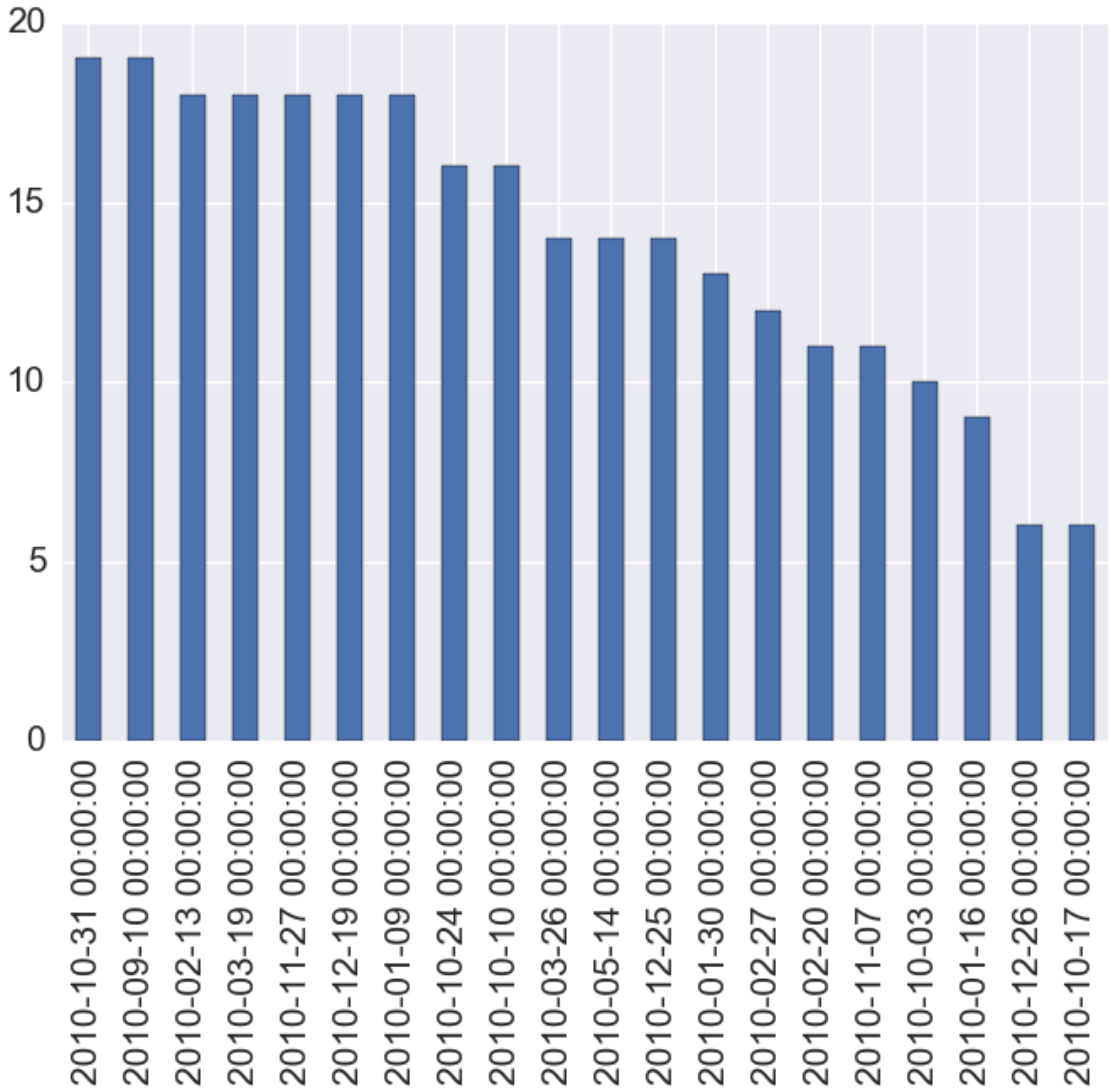


In [15]:

```
mydata['Date'].value_counts().tail(20).plot(kind='bar')
```

Out[15]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x119df01d0>
```



In [16]:

```
count_day = mydata.groupby('Date').count()
count_day.head(20)
```

Out[16]:

	Recordnum	Cardnum	Merchantnum	Merch Description	Merchant State	Merchant Zip	Transty
Date							

Date							
2010-01-01	50	50	50	50	50	50	50
2010-01-02	29	29	29	29	29	10	29
2010-01-03	158	158	152	158	156	152	158
2010-01-04	228	228	220	228	225	220	228
2010-01-05	309	309	286	309	299	297	309
2010-01-06	330	330	317	330	328	321	330
2010-01-07	305	305	290	305	303	292	305
2010-01-08	104	104	104	104	104	104	104
2010-01-09	18	18	18	18	18	18	18
2010-01-10	322	322	306	322	313	308	322
2010-01-11	318	318	302	318	316	310	318
2010-01-12	442	442	426	442	433	422	442
2010-01-13	373	373	364	373	370	352	373
2010-01-14	350	350	339	350	347	332	350
2010-01-15	138	138	135	138	138	135	138
2010-01-16	9	9	9	9	9	9	9
2010-01-17	177	177	172	177	172	173	177
2010-01-18	234	234	229	234	233	222	234

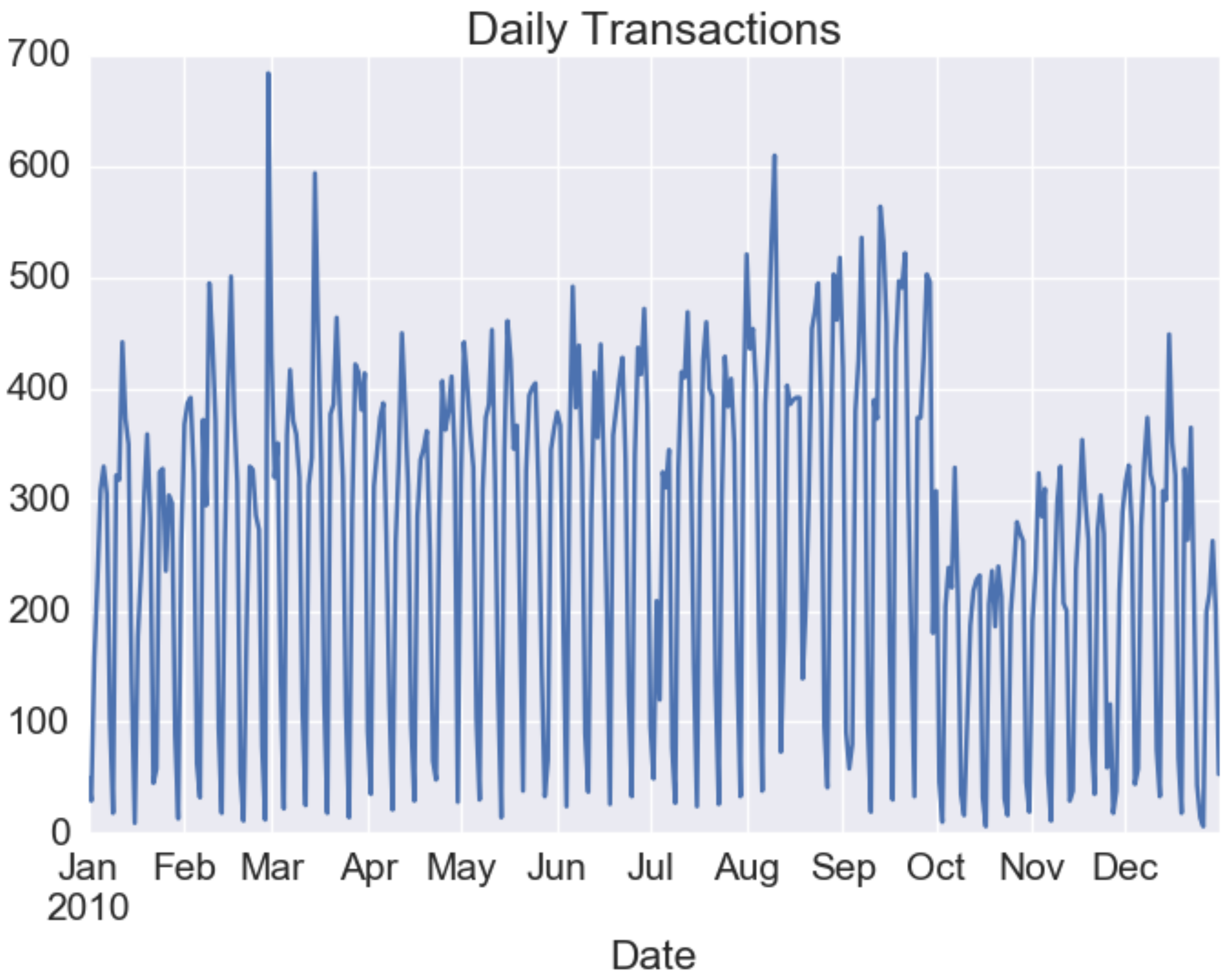
2010-01-19	301	301	295	301	300	264	301
2010-01-20	359	359	322	359	334	310	359

In [17]:

```
mydata.assign(trx = np.ones(numrecords)).set_index(mydata['Date']).resample(dt.timed
    .count().trx.plot(title = 'Daily Transactions')
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x118776e80>
```

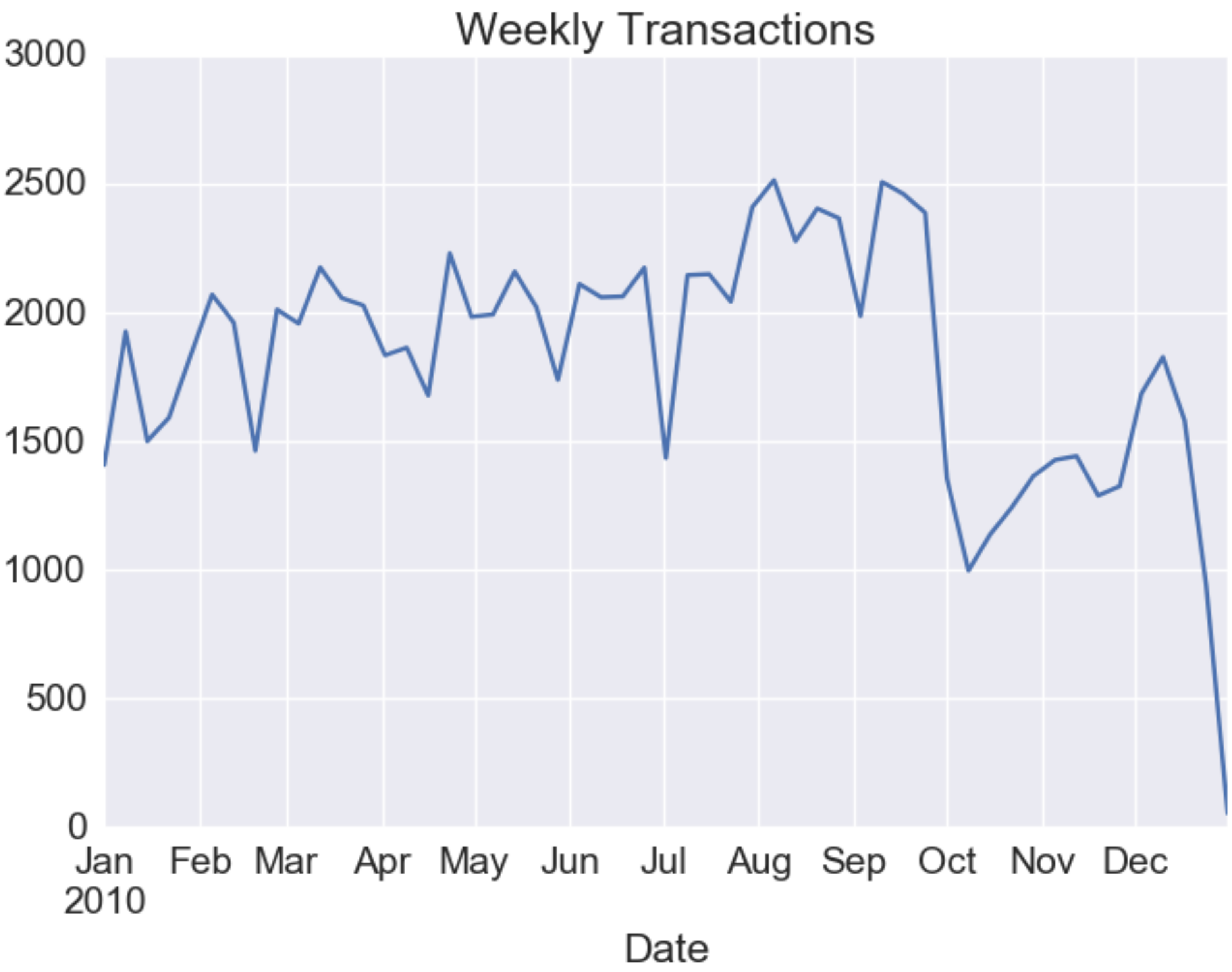


```
In [18]:
```

```
mydata.assign(trx = np.ones(numrecords)).set_index(mydata['Date']).resample(dt.timed
    .count().trx.plot(title = 'Weekly Transactions')
```

```
Out[18]:

<matplotlib.axes._subplots.AxesSubplot at 0x118352b38>
```

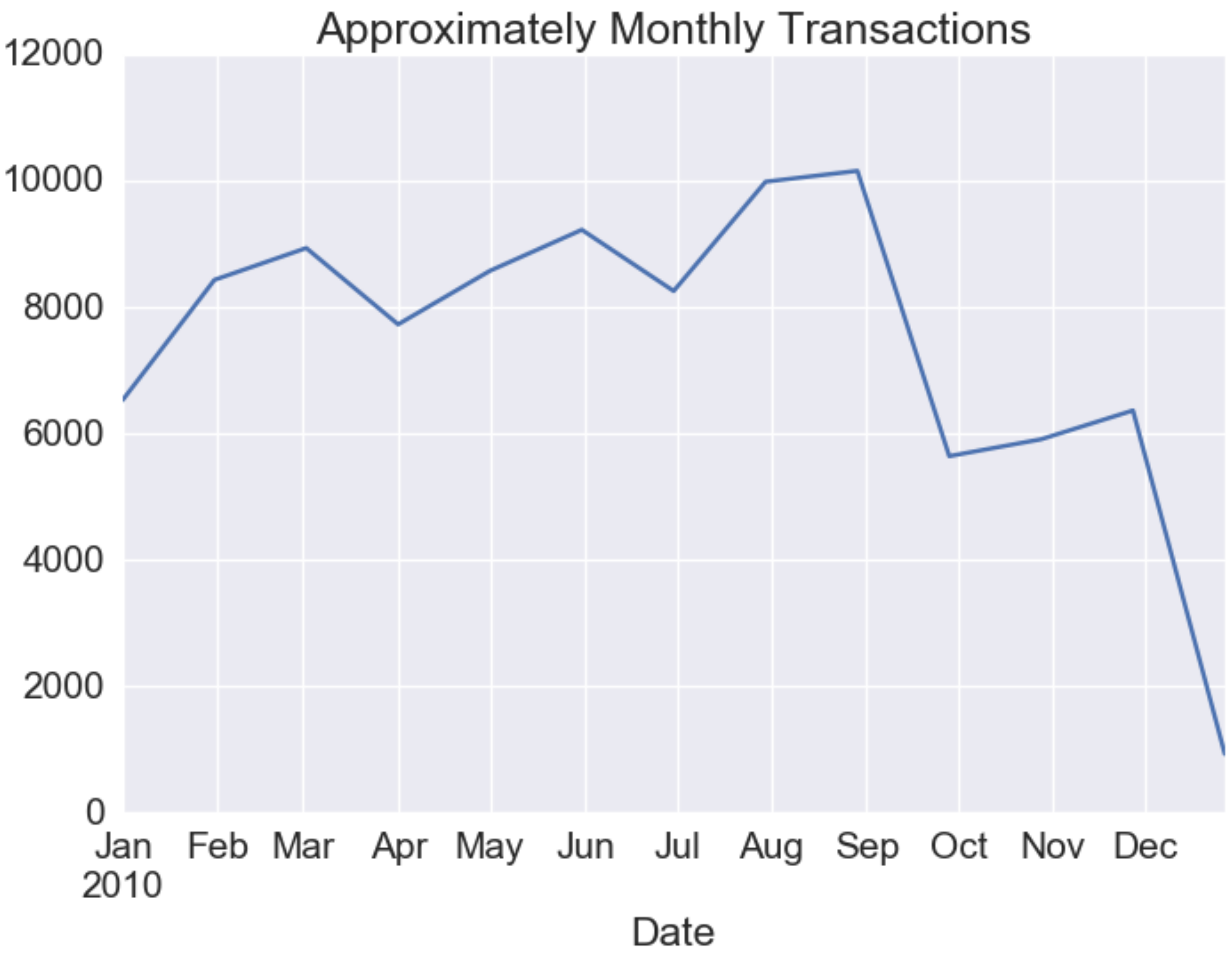


In [19]:

```
mydata.assign(trx = np.ones(numrecords)).set_index(mydata['Date']).resample(dt.timed
    .count().trx.plot(title = 'Approximately Monthly Transactions')
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11a085cf8>
```



In [20]:

```
len(mydata['Merchantnum'].unique())
```

Out[20]:

```
13091
```

In [21]:

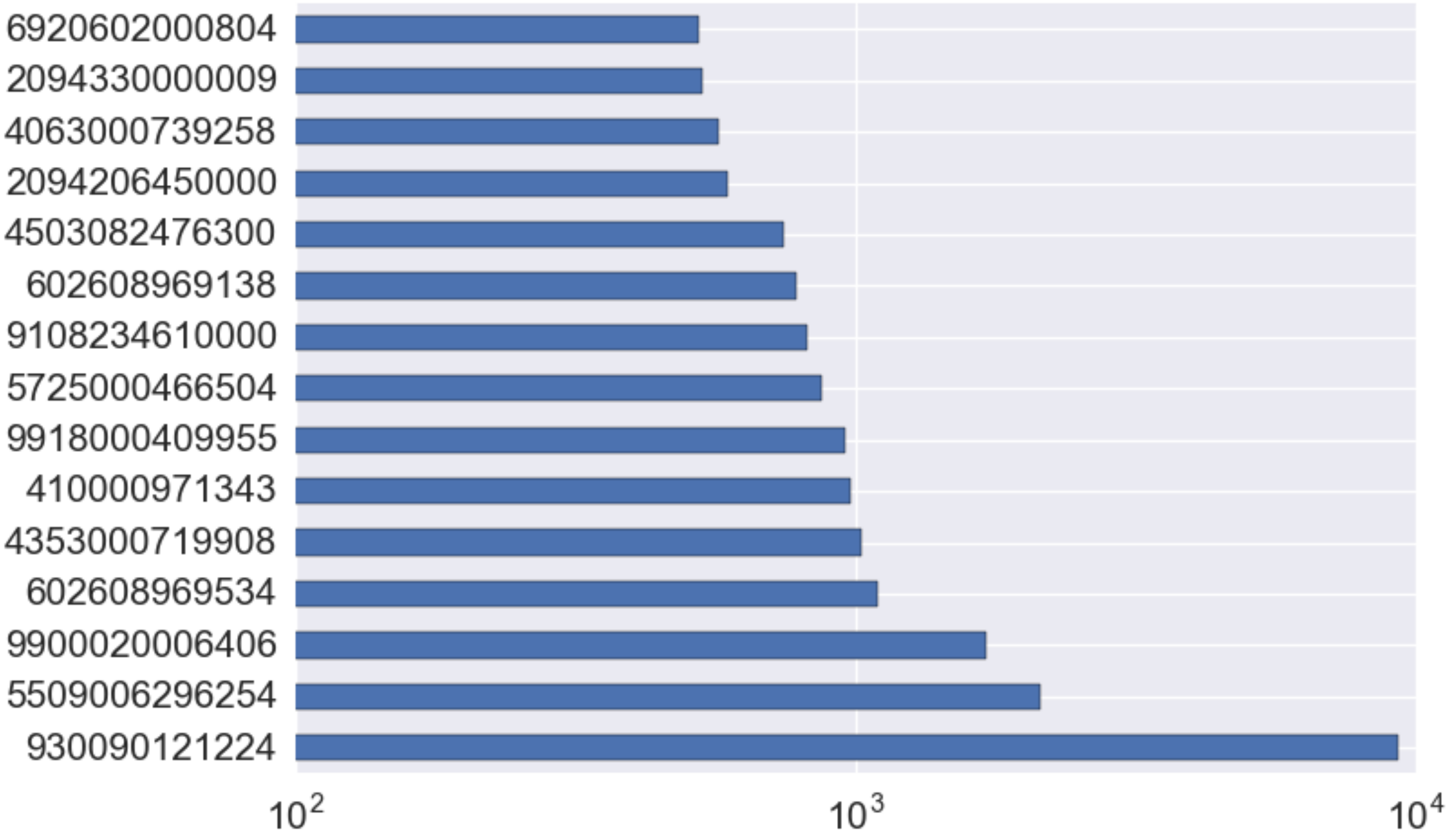
```
mydata['Merchantnum'].value_counts().head(10)
```

Out[21]:

```
930090121224     9310
5509006296254    2131
9900020006406    1714
602608969534     1092
4353000719908    1020
410000971343      982
9918000409955     956
5725000466504     872
9108234610000     817
602608969138      783
Name: Merchantnum, dtype: int64
```

In [22]:

```
mydata['Merchantnum'].value_counts().head(15).plot(kind = 'barh')
plt.xscale('log')
```



In [23]:

```
len(mydata['Merch Description'].unique())
```

Out[23]:

```
13125
```

In [24]:

```
mydata['Merch Description'].value_counts()
```

Out[24]:

```
GSA-FSS-ADV                    1688
SIGMA-ALDRICH                  1635
STAPLES #941                   1174
FISHER SCI ATL                 1093
MWI*MICRO WAREHOUSE             958
CDW*GOVERNMENT INC              872
DELL MARKETING L.P.             816
FISHER SCI CHI                  783
AMAZON.COM  *SUPERSTOR          750
OFFICE DEPOT #1082              748
VWR SCIENTIFIC PROD VCTS        688
PC *PC CONNECTION               570
C & C PRODUCT SERVICES          558
BUY.COM                         481
FISHER SCI HUS                  442
GSA/CUST SUPPLY CTR 97          435
LAB SAFETY SUPPLY, INC          431
PROFESS OFC ENTERPRISES         421
FRANKLIN COVEY COMPANY          418
STAPLES NATIONAL #471           417
GLOBAL COMPUTER SUPPLY          410
A DAIGER AND CO INC             392
RETAIL CREDIT ADJUSTMENT        383
GOVERNMENT SCIENTIFIC SOU       362
LABSOURCE INC                   346
COLE PARMER INSTRUMENT          341
MC MASTER CARR SUPP             311
GTSI                            309
RETAIL DEBIT ADJUSTMENT         308
THE LIGHTHOUSE                  307
                               ...
GRAINGER #753                     1
UOH-VICTORIA VIP                  1
CABLELAN PRODUCTS INC             1
PLANALTAO LOJA 13                 1
THE SIGN POST                     1
MANNSVILLE CHM73690018            1
MICHAELS STORES, INC. #95         1
CU *CONSUMER REPORTS              1
```

```
GREENVIEW DATA INC              1
AMES DEPT STOR 0007377          1
INTERNET PICTURES CORP          1
OFFICE MAX    00009738          1
MOUNTAIN SPORTS                 1
PLUMBING SUPPLY GROUP           1
MARYLAND OFFICE PLANNING        1
WOODS & POOLE ECONOMIC          1
HOME DEPOT #115                 1
INETCAM INC                     1
BRUNO'S #231                    1
ESSENTIAL DATA INC              1
FRAME IT YOURSELF               1
BOISECASCADE*IN#677530          1
HAVANA AUTO PARTS               1
NTHP                            1
DIGITAL IMAGE MANAGEMENT        1
COMPUSA #335                    1
PBS*PUBLIC BROADCASTIN          1
WESTERN POLY DRUMS,INC          1
THE COORDIATE                   1
INDEPENDENT PHOTO ART SPL       1
Name: Merch Description, dtype: int64
```

In [25]:

```
len(mydata['Merchant State'].unique())
```

Out[25]:

```
228
```

In [26]:

```
mydata['Merchant State'].value_counts()
```

Out[26]:

```
TN     11990
VA      7872
CA      6817
IL      6508
MD      5398
GA      5025
PA      4899
NJ      3912
TX      3790
NC      3322
WA      3300
DC      3208
OH      3131
NY      2430
MO      2420
```

```
FL      2143
MA      2081
MI      2033
CO      1987
OR      1510
KS      1236
WI       953
CT       952
MN       939
UT       939
NH       908
NV       726
KY       520
RI       467
OK       411
       ...
971        1
499        1
541        1
293        1
438        1
US         1
391        1
346        1
269        1
769        1
180        1
296        1
390        1
586        1
117        1
460        1
580        1
885        1
705        1
125        1
554        1
497        1
480        1
559        1
619        1
147        1
477        1
870        1
411        1
759        1
Name: Merchant State, dtype: int64
```

In [27]:

```
len(mydata['Merchant Zip'].unique())
```

Out[27]:

```
4568
```

In [28]:

```
mydata['Merchant Zip'].value_counts()
```

Out[28]:

```
38118.0    11823
63103.0     1650
8701.0      1267
22202.0     1250
60061.0     1221
98101.0     1197
17201.0     1180
30091.0     1092
60143.0      942
60069.0      826
78682.0      817
19380.0      769
20763.0      749
20005.0      648
20748.0      592
20151.0      588
22182.0      583
97213.0      578
22304.0      563
92656.0      552
20036.0      522
84119.0      513
22150.0      501
77251.0      487
19103.0      477
53546.0      432
7606.0       419
22314.0      400
60610.0      373
27707.0      362
            ...
43604.0        1
43623.0        1
44039.0        1
44073.0        1
44074.0        1
44077.0        1
44140.0        1
44142.0        1
```

```
44144.0        1
44210.0        1
44302.0        1
44319.0        1
44333.0        1
44451.0        1
44667.0        1
44675.0        1
44706.0        1
45041.0        1
45106.0        1
45204.0        1
45217.0        1
45232.0        1
45356.0        1
45365.0        1
45406.0        1
45446.0        1
45449.0        1
45479.0        1
45504.0        1
44503.0        1
Name: Merchant Zip, dtype: int64
```

In [29]:

```
mydata['Transtype'].value_counts()
```

Out[29]:

```
P    96353
A      181
D      173
Y        1
Name: Transtype, dtype: int64
```

In [30]:

```
mydata['Amount'].value_counts()
```

Out[30]:

```
3.62       4283
3.67       1620
3.74        913
3.80        827
4.37        378
30.00       317
3.85        271
100.00      252
75.00       243
6.62        219
19.95       210
```

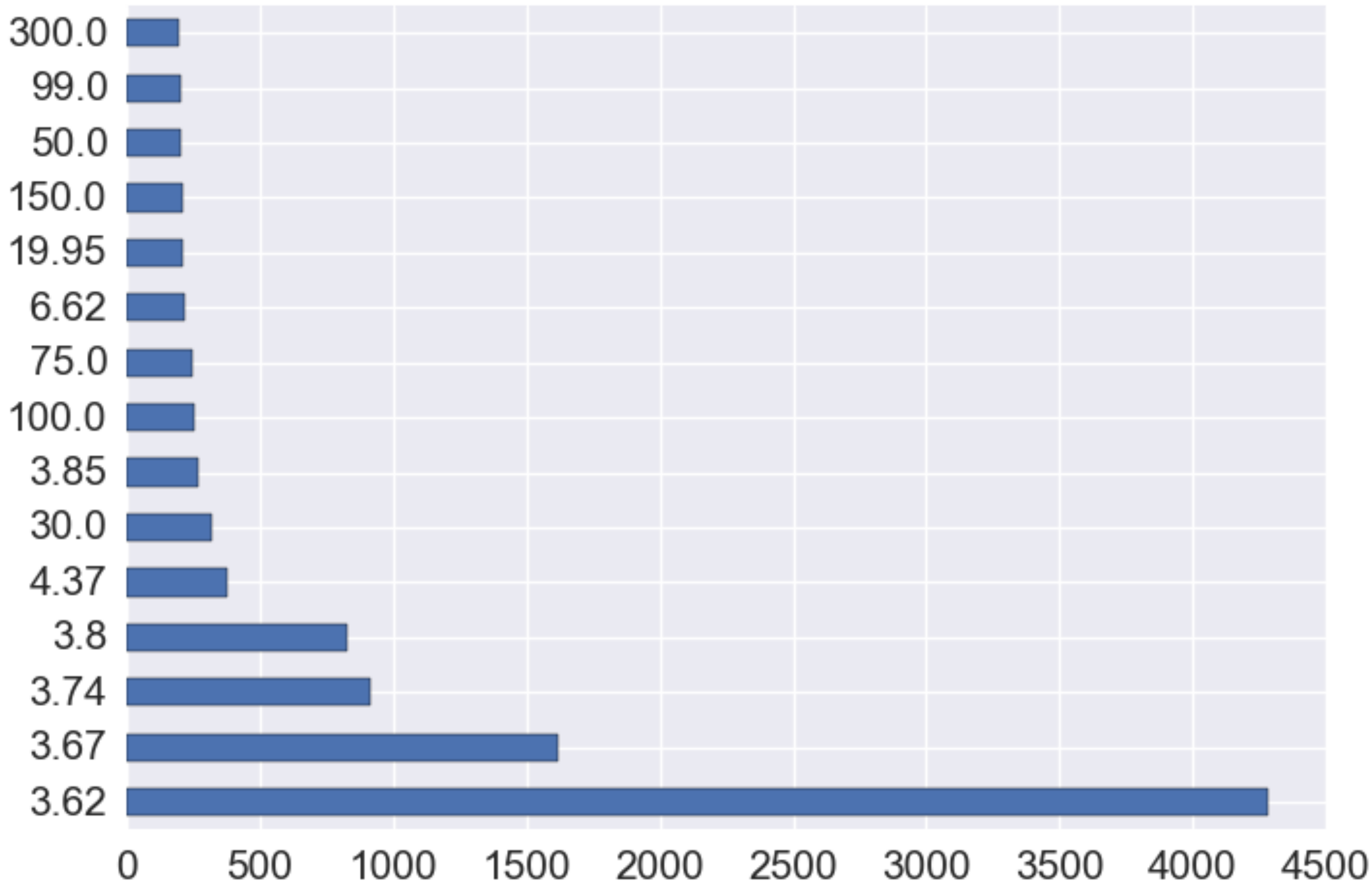
```
150.00      208
50.00       205
99.00       200
300.00      196
200.00      193
350.00      178
25.00       171
250.00      171
8.31        164
60.00       159
295.00      158
2500.00     157
35.00       157
195.00      156
500.00      149
20.00       147
199.00      146
3.57        136
125.00      135
           ... 
180.49        1
92.37         1
179.26        1
257.63        1
253.87        1
491.78        1
487.74        1
486.01        1
485.76        1
28.14         1
260.39        1
510.21        1
134.55        1
172.26        1
482.99        1
30.89         1
481.76        1
261.85        1
480.72        1
173.74        1
32.39         1
262.90        1
135.32        1
33.64         1
495.99        1
263.40        1
494.47        1
178.99        1
132.44        1
0.50          1
Name: Amount, dtype: int64
```

In [31]:

```
mydata['Amount'].value_counts().head(15).plot(kind = 'barh')
```

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11a0922e8>
```

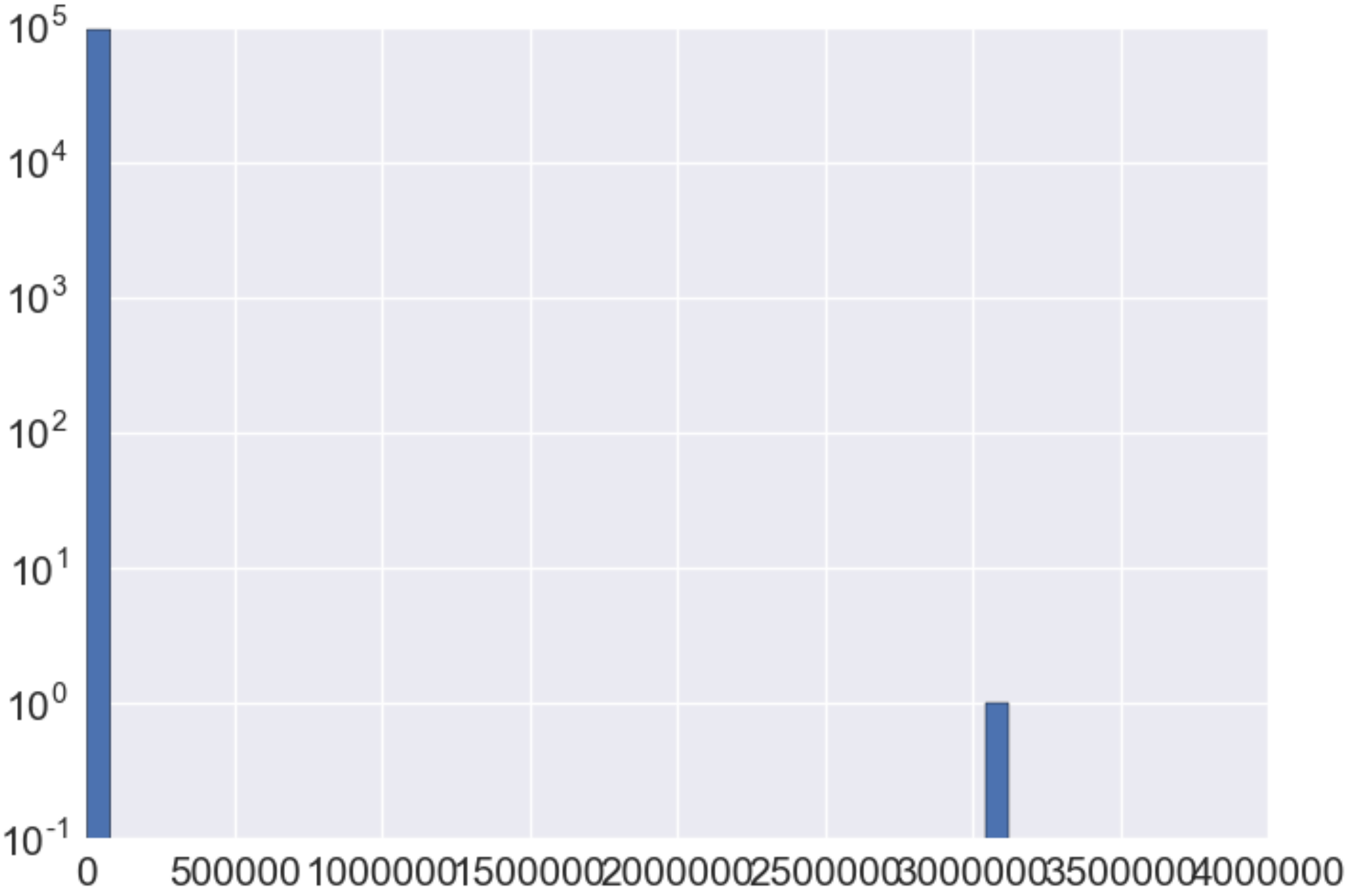


In [32]:

```
plt.hist(mydata['Amount'],bins=50,range=[0,4000000])
plt.yscale('log')
plt.ylim(ymin=.1)
```

Out[32]:

```
(0.1, 100000.0)
```

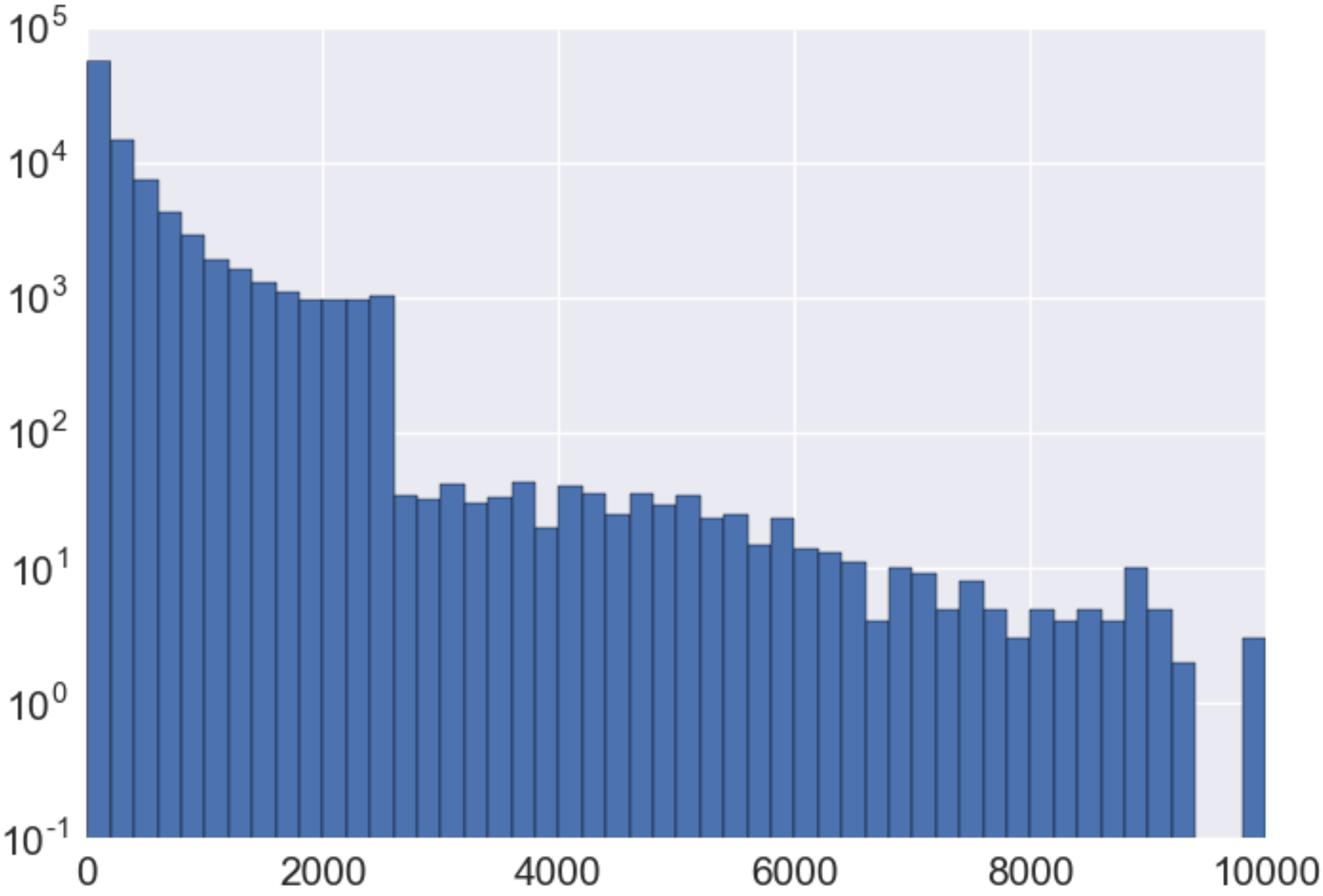


In [33]:

```
plt.hist(mydata['Amount'],bins=50,range=[0,10000])
plt.yscale('log')
plt.ylim(ymin=.1)
```

Out[33]:

```
(0.1, 100000.0)
```

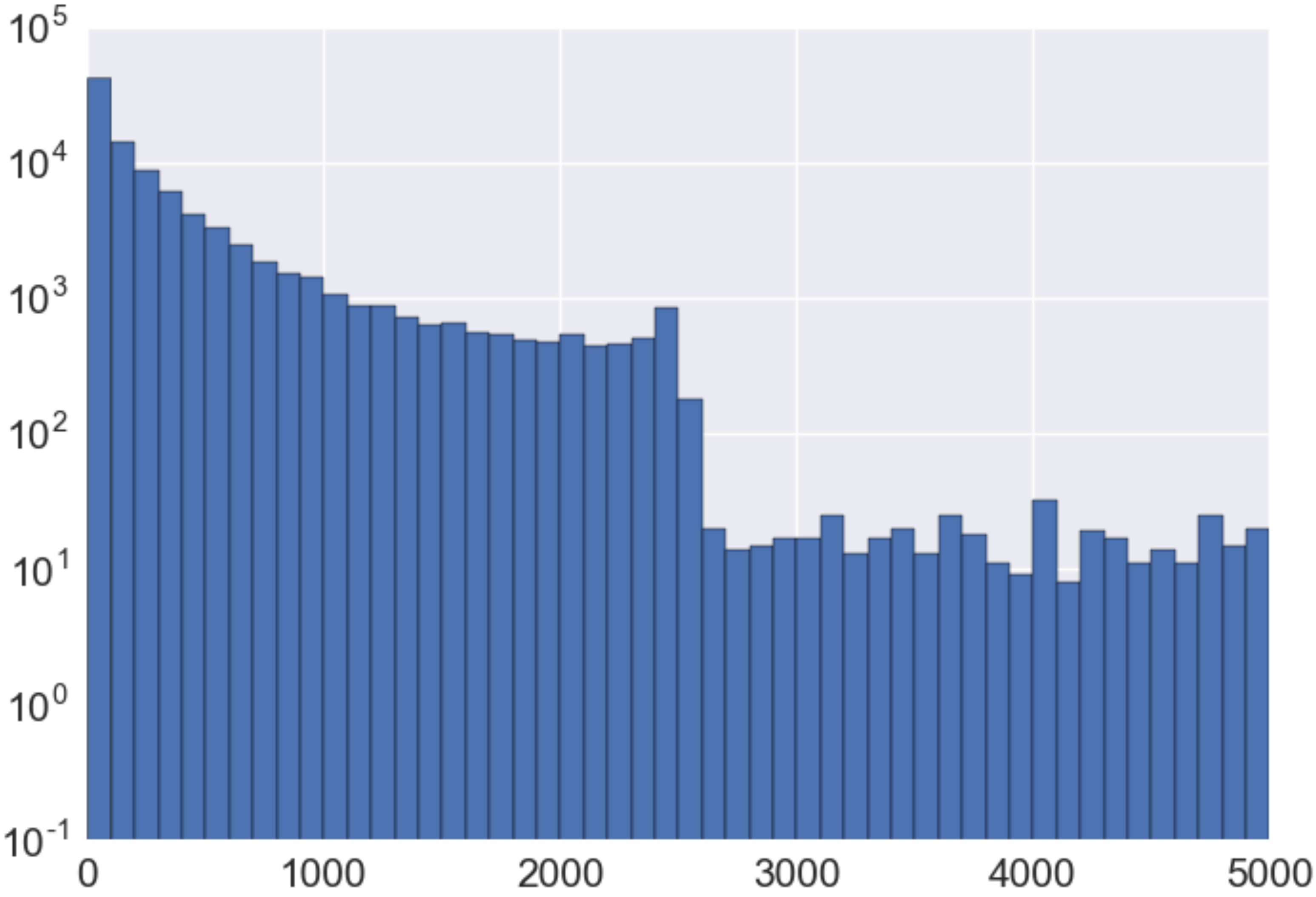


In [34]:

```
plt.hist(mydata['Amount'],bins=50,range=[0,5000])
plt.yscale('log')
plt.ylim(ymin=.1)
```

Out[34]:

```
(0.1, 100000.0)
```

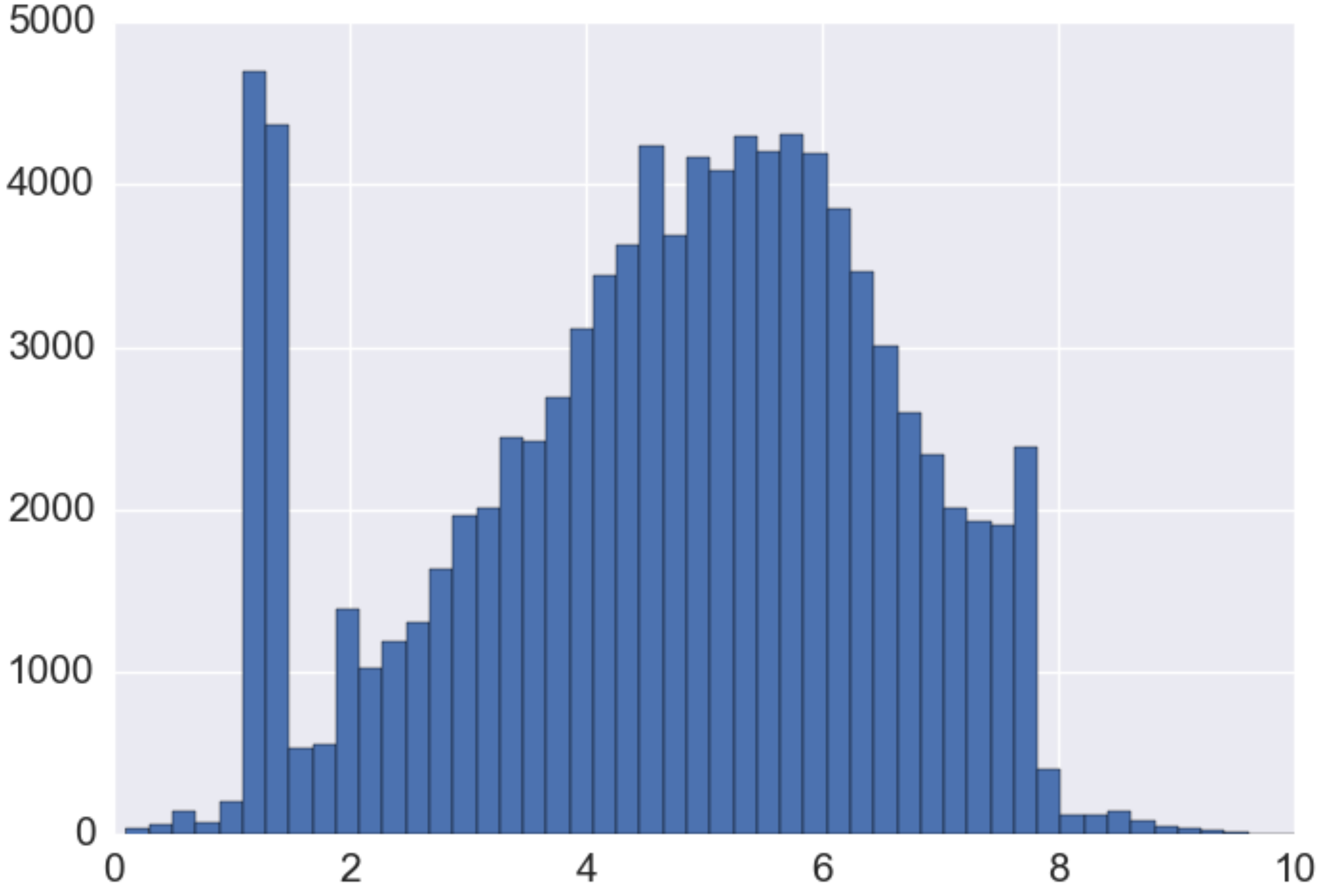


In [35]:

```
mydata['logamount'] = np.log(mydata['Amount'])
plt.hist(mydata['logamount'], bins=50,range=[.1, 10])
```

Out[35]:

```
(array([  38.,    60.,   142.,    75.,   198., 4694., 4367.,  527.,  551.,
        1391., 1024., 1184., 1301., 1630., 1966., 2011., 2438., 2414.,
        2685., 3107., 3437., 3626., 4233., 3686., 4171., 4090., 4294.,
        4203., 4308., 4194., 3846., 3468., 3005., 2600., 2342., 2012.,
        1925., 1902., 2388.,  407.,  125.,  125.,  148.,   86.,   46.,
          33.,   24.,   16.,    9.,    5.]),
 array([ 0.1  ,  0.298,  0.496,  0.694,  0.892,  1.09 ,  1.288,  1.486
,
         1.684,  1.882,  2.08 ,  2.278,  2.476,  2.674,  2.872,  3.07 
,
         3.268,  3.466,  3.664,  3.862,  4.06 ,  4.258,  4.456,  4.654
,
         4.852,  5.05 ,  5.248,  5.446,  5.644,  5.842,  6.04 ,  6.238
,
         6.436,  6.634,  6.832,  7.03 ,  7.228,  7.426,  7.624,  7.822
,
         8.02 ,  8.218,  8.416,  8.614,  8.812,  9.01 ,  9.208,  9.406
,
         9.604,  9.802, 10.   ]),
 <a list of 50 Patch objects>)
```



In [36]:

```
mydata['Fraud'].value_counts()
```

Out[36]:

```
0    95694
1     1014
Name: Fraud, dtype: int64
```

In [37]:

```
goods = mydata[mydata['Fraud'] == 0]
bads = mydata[mydata['Fraud'] == 1]
len(goods)
```

Out[37]:

```
95694
```

In [38]:

```
len(bads)
```

Out[38]:

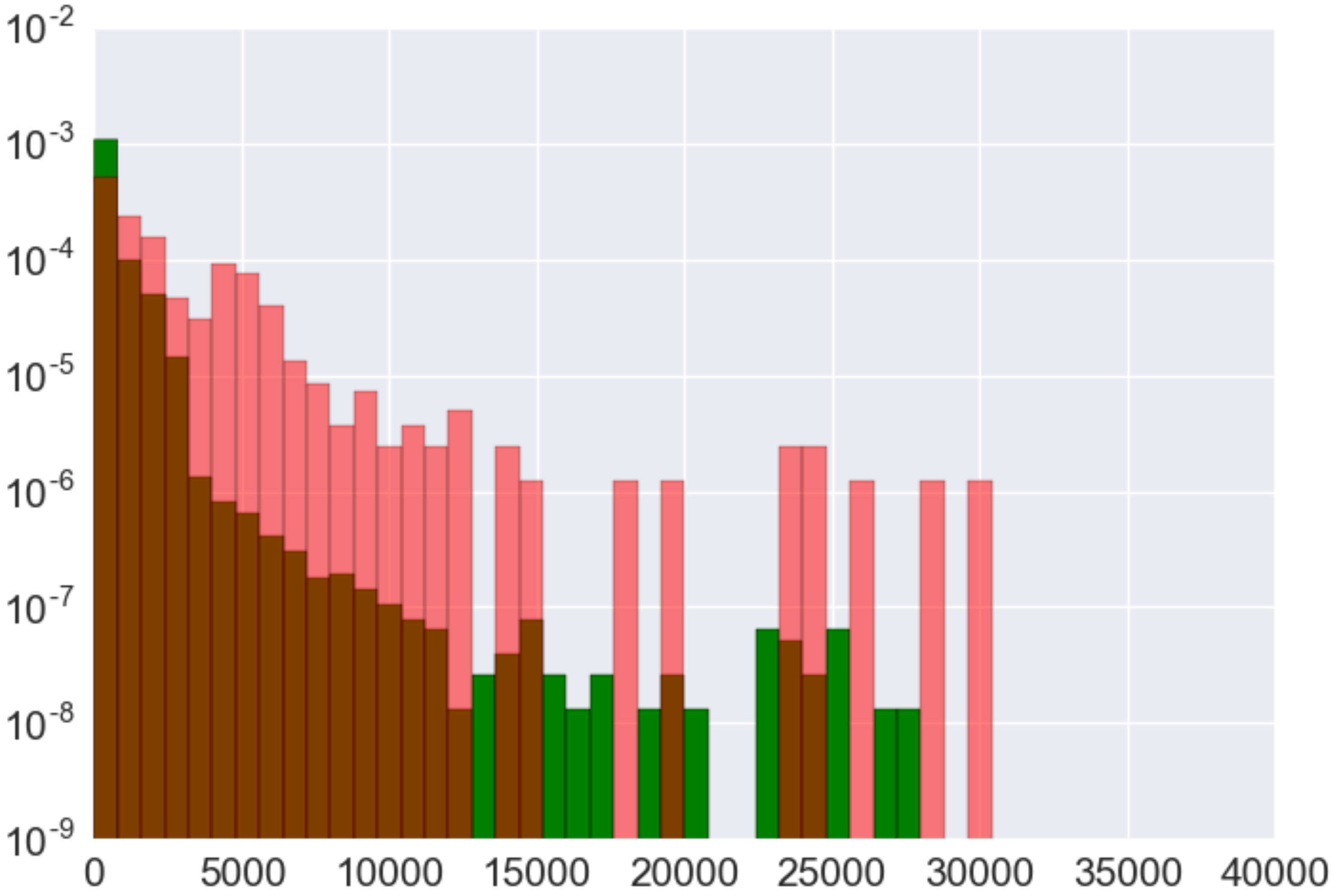
```
1014
```

In [39]:

```
plt.hist(goods['Amount'],bins=50,range=[0,40000], normed = True, color = 'green')
plt.hist(bads['Amount'],bins=50,range=[0,40000], normed = True, color = 'red', alpha
plt.yscale('log')
plt.ylim(ymin=.000000001)
```

Out[39]:

```
(1e-09, 0.01)
```

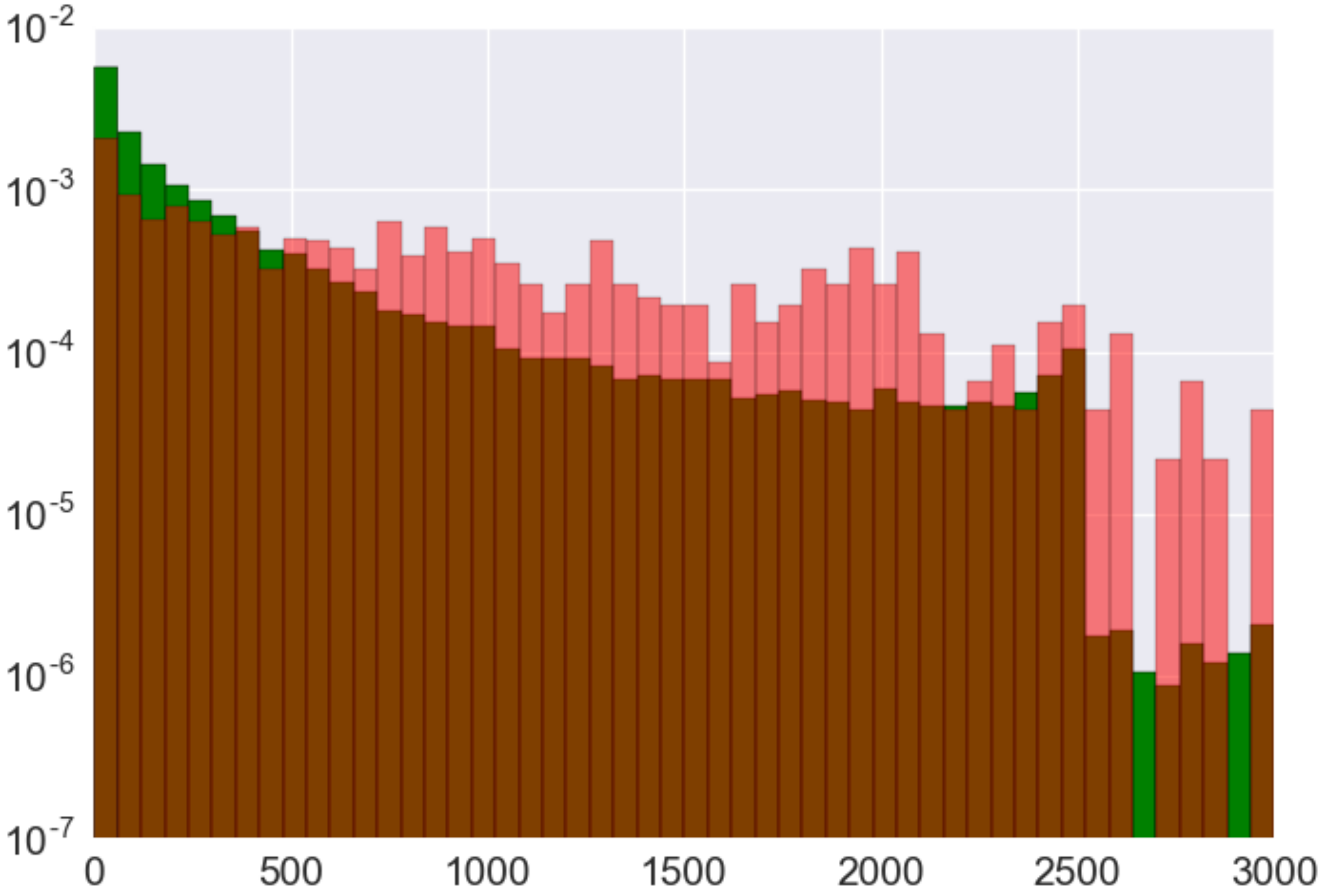


In [40]:

```
plt.hist(goods['Amount'],bins=50,range=[0,3000], normed = True, color = 'green')
plt.hist(bads['Amount'],bins=50,range=[0,3000], normed = True, color = 'red', alpha
plt.yscale('log')
plt.ylim(ymin=.0000001)
```

Out[40]:

```
(1e-07, 0.01)
```

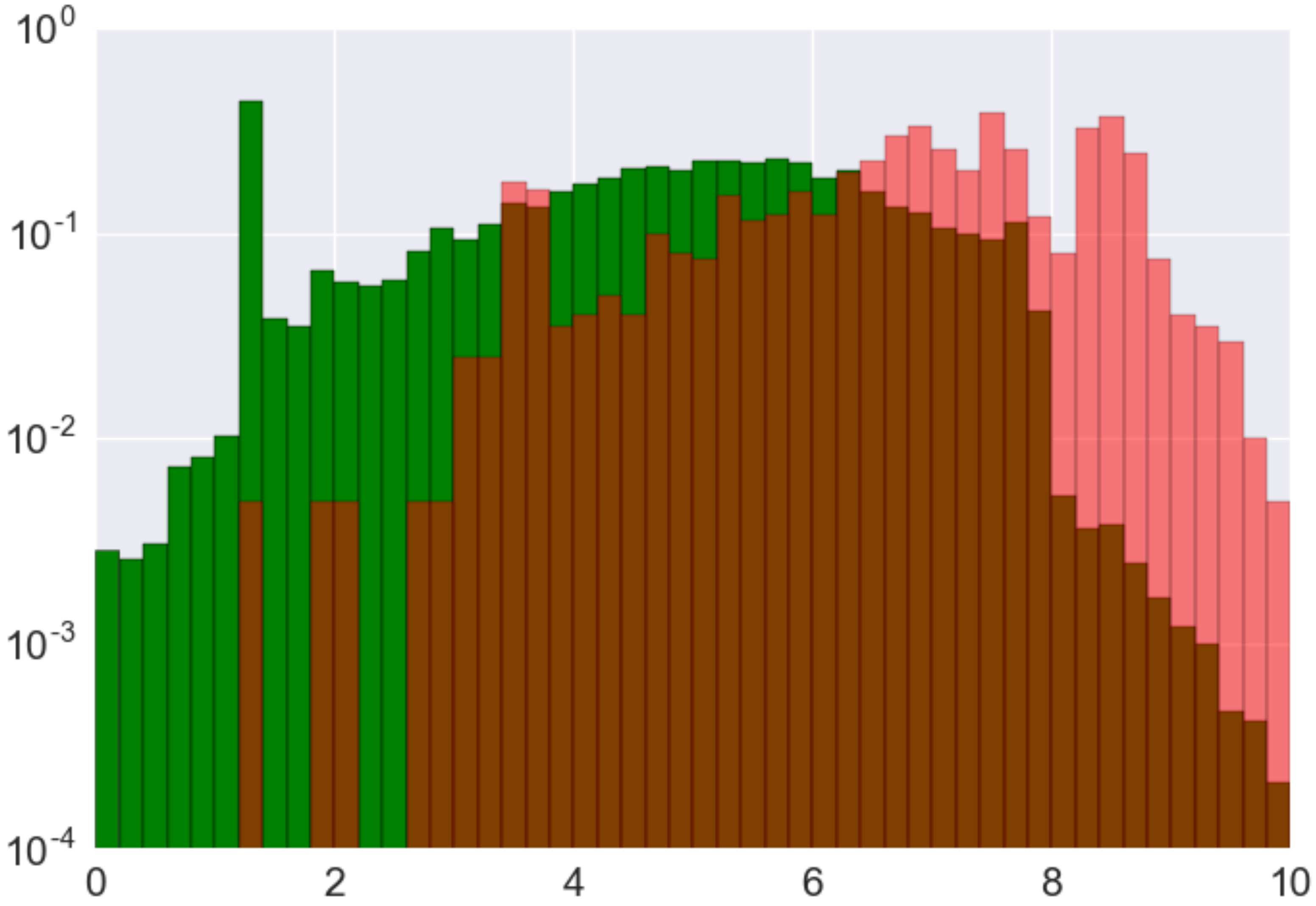


In [41]:

```
plt.hist(goods['logamount'],bins=50,range=[0,10], normed = True, color = 'green')
plt.hist(bads['logamount'],bins=50,range=[0,10], normed = True, color = 'red', alpha
plt.yscale('log')
plt.ylim(ymin=.0001)
```

Out[41]:

```
(0.0001, 1.0)
```

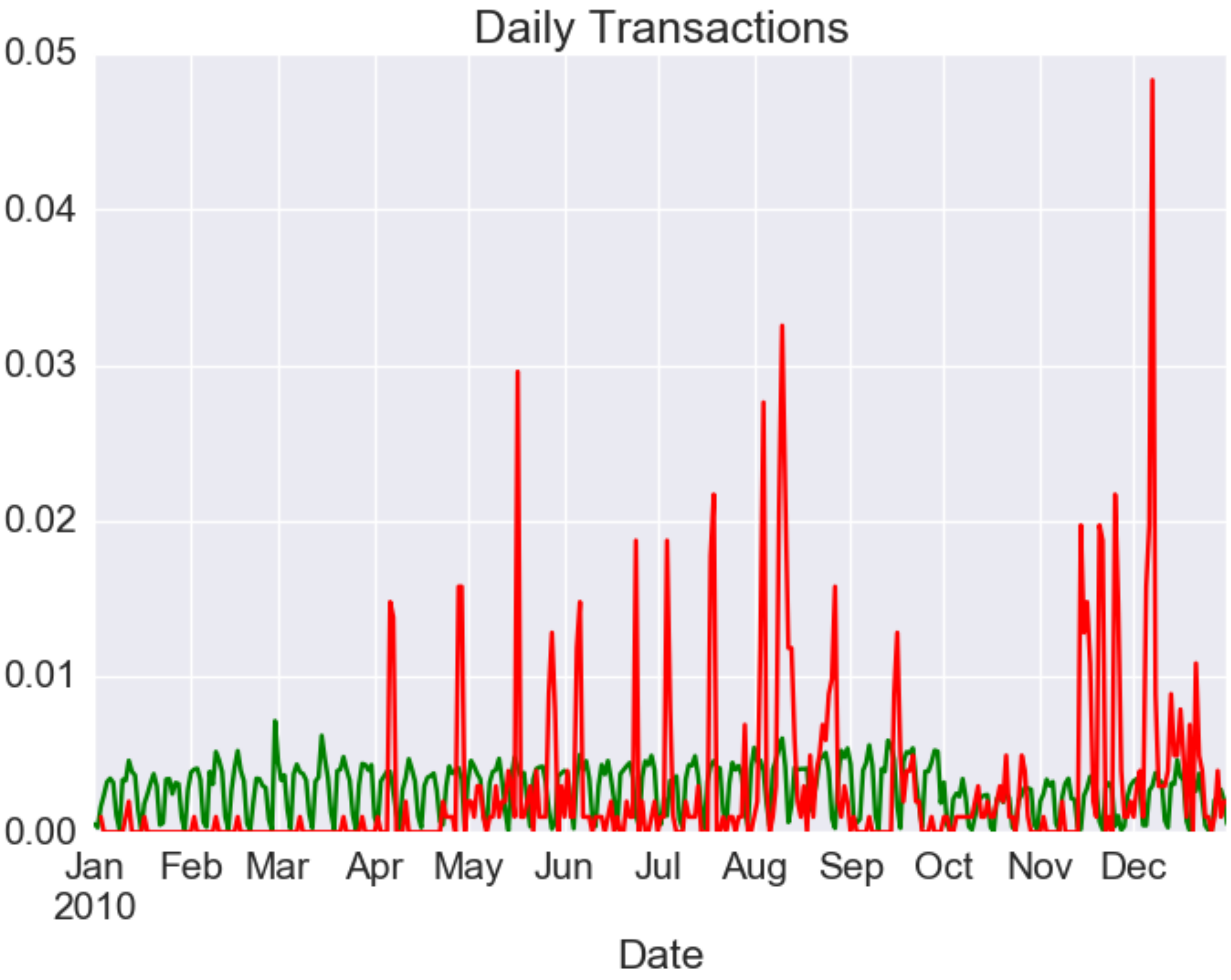


```
In [42]:
```

```
ngoods = len(goods)
nbads = len(bads)
goods_series = goods.assign(trx = np.ones(ngoods)).set_index(goods['Date']).resample
norm_goods_series = goods_series / ngoods
norm_goods_series.plot(title = 'Daily Transactions', color = 'green')
bads_series = bads.assign(trx = np.ones(nbads)).set_index(bads['Date']).resample(dt.
norm_bads_series = bads_series / nbads
norm_bads_series.plot(color = 'red')
```

```
Out[42]:

<matplotlib.axes._subplots.AxesSubplot at 0x11ae30400>
```

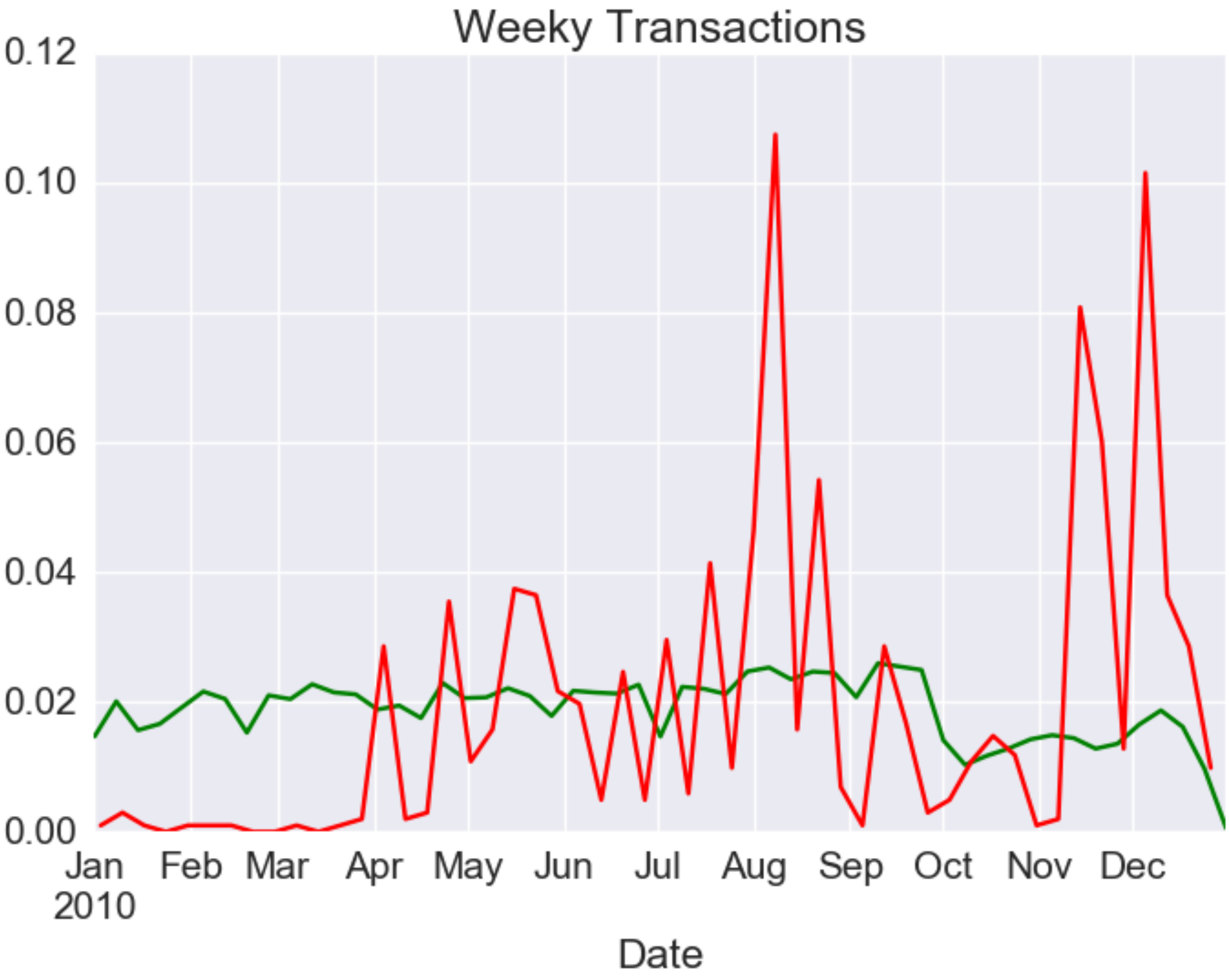


In [43]:

```
goods_series = goods.assign(trx = np.ones(ngoods)).set_index(goods['Date']).resample
norm_goods_series = goods_series / ngoods
norm_goods_series.plot(title = 'Weeky Transactions', color = 'green')
bads_series = bads.assign(trx = np.ones(nbads)).set_index(bads['Date']).resample(dt
norm_bads_series = bads_series / nbads
norm_bads_series.plot(color = 'red')
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x119f0b470>
```

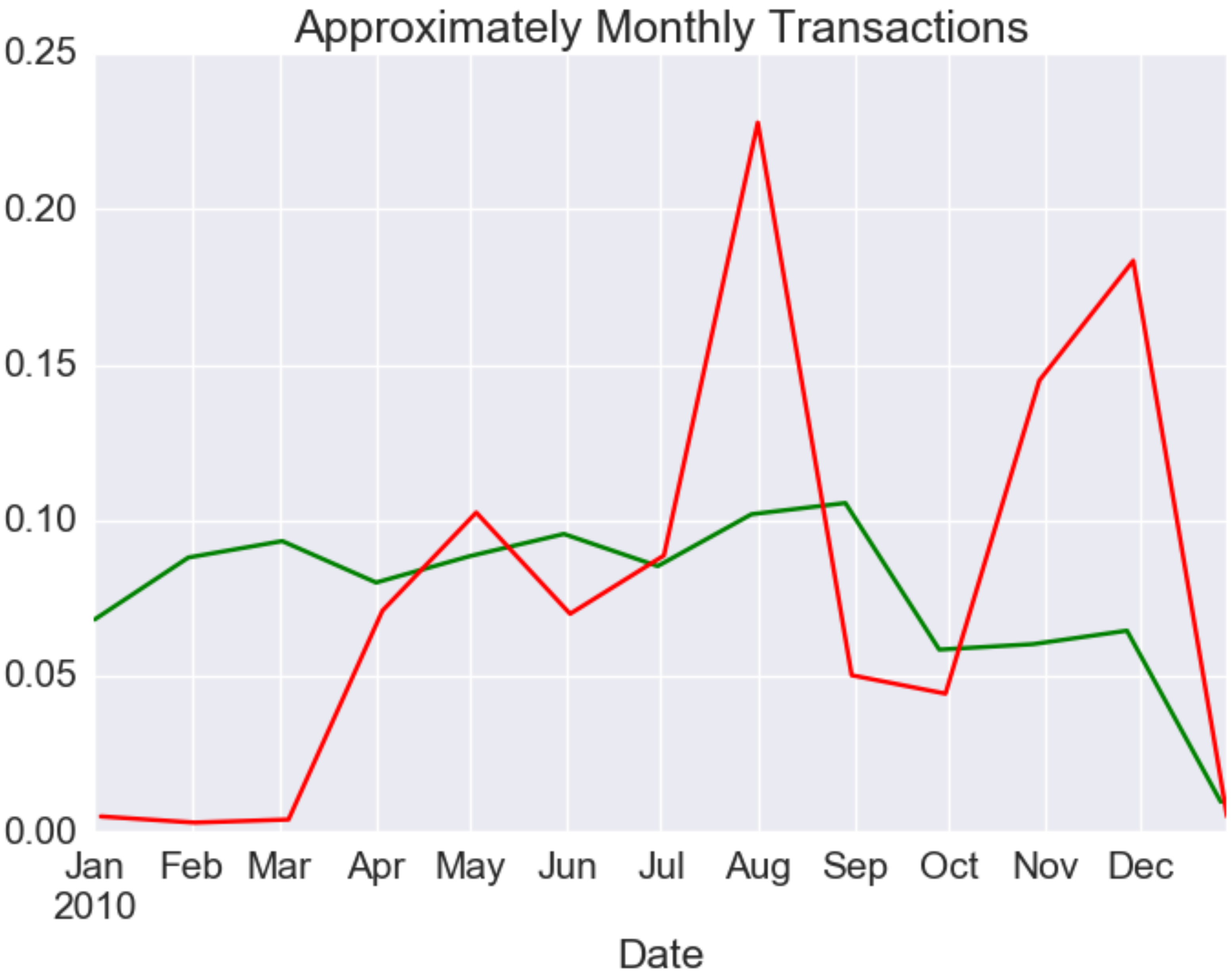


```
In [44]:
```

```
goods_series = goods.assign(trx = np.ones(ngoods)).set_index(goods['Date']).resample
norm_goods_series = goods_series / ngoods
norm_goods_series.plot(title = 'Approximately Monthly Transactions', color = 'green
bads_series = bads.assign(trx = np.ones(nbads)).set_index(bads['Date']).resample(dt.
norm_bads_series = bads_series / nbads
norm_bads_series.plot(color = 'red')
```

```
Out[44]:

<matplotlib.axes._subplots.AxesSubplot at 0x11ac58978>
```



```
In [ ]:
```