In [3]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline
```

1. Data Wrangling

Check each table for values outside of range and missing values.

In [50]:

```
df_trans = pd.read_csv('transactions_v2.csv')
```

In [6]:

```
df_trans.isnull().values.any()
```

Out[6]:

```
False
```

In [23]:

```
df_trans.describe()
```

Out[23]:

	payment_method_id	payment_plan_days	plan_list_price	actual_amount_paid	is_auto_renew	transaction_
count	1.431009e+06	1.431009e+06	1.431009e+06	1.431009e+06	1.431009e+06	1.431009e+(
mean	3.791835e+01	6.601770e+01	2.817870e+02	2.813172e+02	7.853025e-01	2.016848e+(
std	4.964805e+00	1.024864e+02	4.351861e+02	4.354200e+02	4.106124e-01	4.858797e+(
min	2.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.015010e+(
25%	3.600000e+01	3.000000e+01	9.900000e+01	9.900000e+01	1.000000e+00	2.017023e+(
50%	4.000000e+01	3.000000e+01	1.490000e+02	1.490000e+02	1.000000e+00	2.017031e+(
75%	4.100000e+01	3.000000e+01	1.490000e+02	1.490000e+02	1.000000e+00	2.017032e+(
max	4.100000e+01	4.500000e+02	2.000000e+03	2.000000e+03	1.000000e+00	2.017033e+(

Transactions data table doesn't have any missing values and the ranges of the features look good.

In [49]:

```
df_train = pd.read_csv('train_v2.csv')
```

In [18]:

```
df_train.isnull().values.any()
```

Out[18]:

```
False
```

```
In [20]: df_train.describe()
```

Out[20]:

	is_churn
count	970960.000000
mean	0.089942
std	0.286099
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

```
In [4]: df_members = pd.read_csv('members_v3.csv')
```

```
In [39]: df_members.isnull().sum()
```

```
Out[39]: msno                            0
         city                            0
         bd                              0
         gender                    4429505
         registered_via                  0
         registration_init_time          0
         dtype: int64
```

The members table has missing gender values.

```
In [23]: df_members['gender'].isnull().sum()/len(df_members)
```

```
Out[23]: 0.65433527838873129
```

There is a large percentage of users whose gener is unknown, 65%. Since the percentage is so large, I will keep the data points with the missing values.

```
In [5]: df_members.describe()
```

Out[5]:

	city	bd	registered_via	registration_init_time
count	6.769473e+06	6.769473e+06	6.769473e+06	6.769473e+06
mean	3.847358e+00	9.795794e+00	5.253069e+00	2.014518e+07
std	5.478359e+00	1.792590e+01	2.361398e+00	2.318601e+04
min	1.000000e+00	-7.168000e+03	-1.000000e+00	2.004033e+07
25%	1.000000e+00	0.000000e+00	4.000000e+00	2.014042e+07
50%	1.000000e+00	0.000000e+00	4.000000e+00	2.015101e+07
75%	4.000000e+00	2.100000e+01	7.000000e+00	2.016060e+07
max	2.200000e+01	2.016000e+03	1.900000e+01	2.017043e+07

The feature bd gives the users age, and there are some values outside of a range that makes sense. The min age is -7168 and the max age is 2016. The proportion of bad ages is:

```
In [26]: len(df_members[(df_members['bd']<0) | (df_members['bd']>100)])/len(df_members)

Out[26]: 0.0008347769464476777
```

The max value of registered_via should be 16, and the min value should be 3. However, there are values that lie outside of this range. The proportion of bad values is:

```
In [33]: len(df_members[(df_members['registered_via']<3) | (df_members['registered_via']>16)])/len(d
         f_members)

Out[33]: 0.000586308564935557
```

Since the number of users with bad ages is a small percentage of the total users, I will remove these rows.

```
In [6]: df_logs = pd.read_csv('user_logs_v2.csv')
```

```
In [28]: df_logs.describe()
```

Out[28]:

	date	num_25	num_50	num_75	num_985	num_100	num_unq	to
count	1.839636e+07	1.839636e+07	1.839636e+07	1.839636e+07	1.839636e+07	1.839636e+07	1.839636e+07	1.839
mean	2.017032e+07	6.191401e+00	1.508789e+00	9.413759e-01	1.079905e+00	3.028246e+01	2.903615e+01	7.904
std	8.916720e+00	1.342827e+01	3.908539e+00	1.924840e+00	3.518409e+00	4.203641e+01	3.219866e+01	1.013
min	2.017030e+07	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00	1.000
25%	2.017031e+07	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	7.000000e+00	8.000000e+00	1.959
50%	2.017032e+07	2.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	1.700000e+01	1.800000e+01	4.582
75%	2.017032e+07	7.000000e+00	2.000000e+00	1.000000e+00	1.000000e+00	3.700000e+01	3.800000e+01	9.848
max	2.017033e+07	5.639000e+03	9.120000e+02	5.080000e+02	1.561000e+03	4.110700e+04	4.925000e+03	9.194

```
In [30]: df_logs.isnull().values.any()

Out[30]: False
```

The logs table doesn't have any null values, and the ranges for the features seems sensible.

2. Data Storytelling

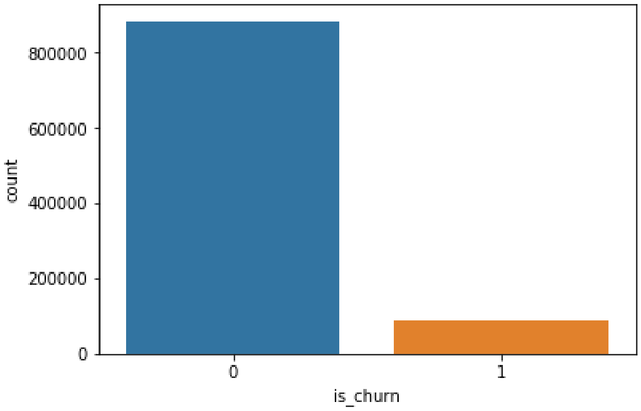
```
In [51]: len(df_train[df_train.is_churn==1])/len(df_train)

Out[51]: 0.08994191315811156
```

About 10% of the users have churned.

```
In [7]: sns.countplot(x='is_churn',data=df_train)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7f10288a51d0>
```



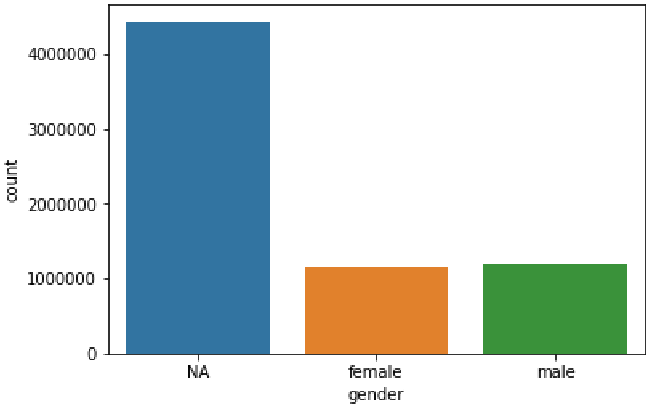
The majority of the users did not list their genders. Of the users who did list their gender, the number of male users is almost equal to the number of female users.

```
In [10]: df_members_na = df_members.fillna('NA')
```

```
In [11]: sns.countplot(x='gender',data=df_members_na)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0ffe7185f8>
```

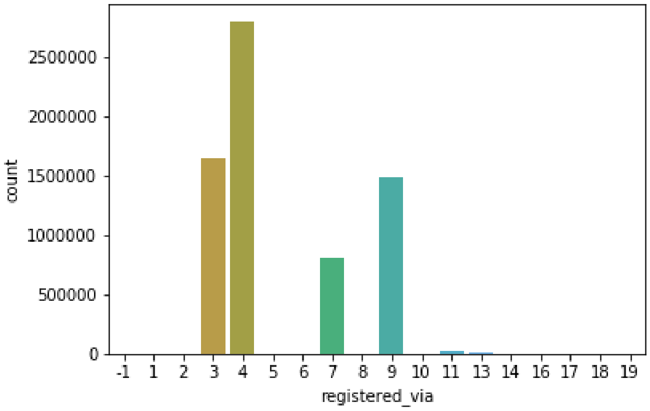


The most popular registration method is method 4. Methods 3 and 9 are the next most popular, followed by method 7, about half as many signups as 3 & 9). The rest of the methods are not very popular.

```
In [13]: sns.countplot(x='registered_via',data=df_members_na)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0ffe4f5be0>
```

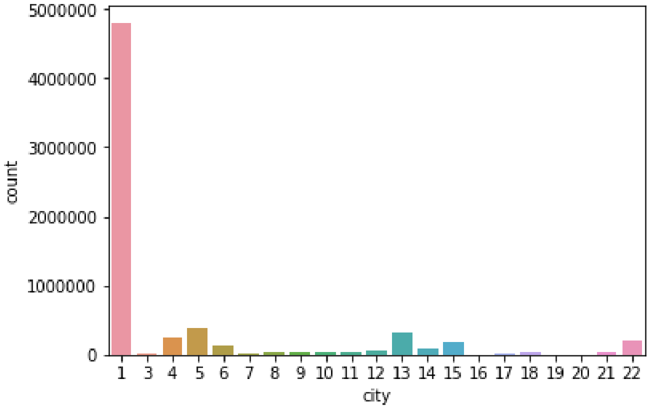


Most of the users live in the city labeled 1. The other 21 cities do not have many registrations.

```
In [18]: sns.countplot(x='city',data=df_members_na)
```

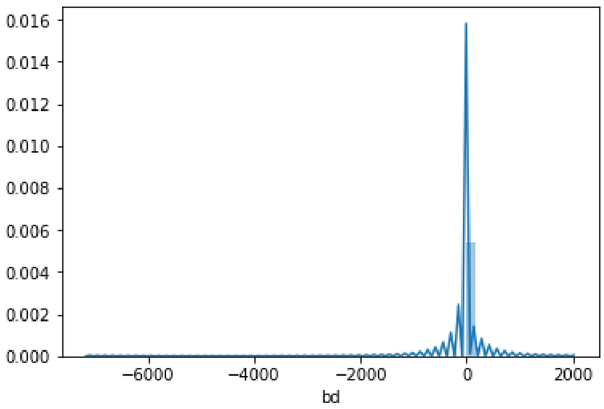
```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0ffe0650b8>
```



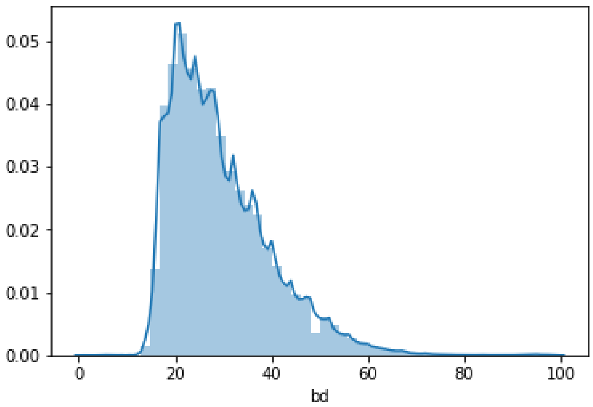
As we saw before, there are many incorrect values for user age.

```
In [23]: sns.distplot(df_members['bd'])

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0ffe0c7550>
```



Plotting only sensible values for the age, shows that most of the users are teenageers and young adults. The distribution of ages peaks around 25, and then decreases.

```
In [26]: sns.distplot(df_members['bd'][(df_members.bd>0) & (df_members.bd<100)])

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0ffdcd3cc0>
```

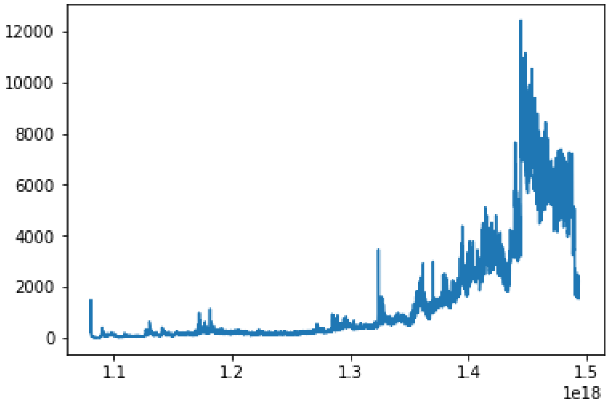


```
In [5]: df_members['registration_init_time'] = pd.to_datetime(df_members['registration_init_time'],
         format='%Y%m%d')
```

The service wasn't very popular until 2010, at which point signups began to slowly increase. After 2015, signups increased strongly. However, recently new registrations have been declining.

```
In [17]: plt.plot(df_members['registration_init_time'].value_counts().sort_index())
```

```
Out[17]: [<matplotlib.lines.Line2D at 0x7f1e69c067f0>]
```

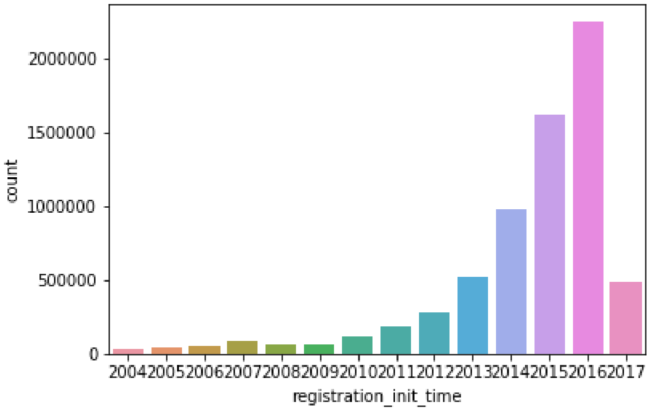


```
In [44]: year = pd.to_datetime(df_members['registration_init_time'], format='%Y%m%d').dt.year
         year = year.to_frame()
```

```
In [46]: sns.countplot(x='registration_init_time',data=year)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[46]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1e6763f6d8>
```



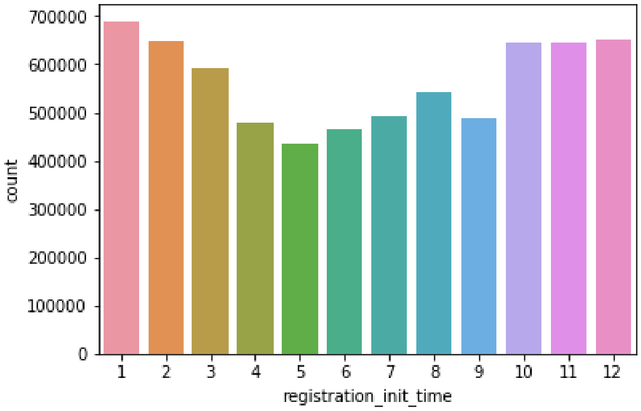
```
In [40]: months = pd.to_datetime(df_members['registration_init_time'], format='%Y%m%d').dt.month
         months=months.to_frame()
```

The beginning of the year and the end of the year have higher initial registrations than the other months. The least popular months are April and May. Note that this trend might not hold over all years.

```
In [43]: sns.countplot(x='registration_init_time',data=months)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1e676bb320>
```



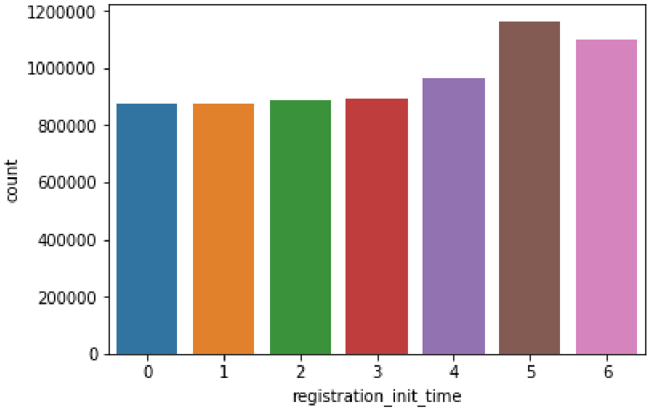
```
In [45]: dayofweek = pd.to_datetime(df_members['registration_init_time'], format='%Y%m%d').dt.dayofw
         eek
         dayofweek = dayofweek.to_frame()
```

Weekend days have higher initial registrations than the rest of the week. The weekday with the most registrations is Friday.

```
In [47]: sns.countplot(x='registration_init_time',data=dayofweek)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1e675d7e80>
```

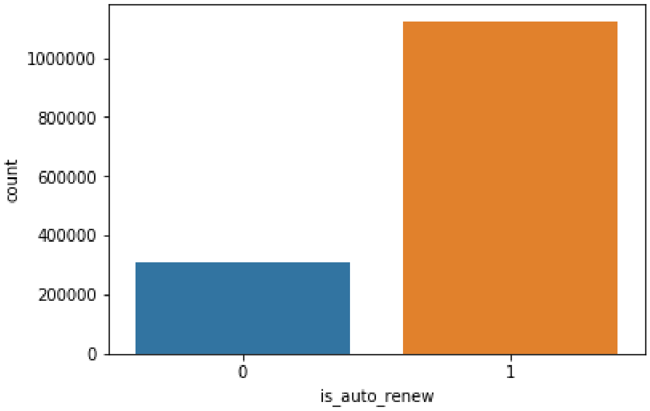


Looking at the mean of is_auto_renew, we see that the majority of users, 78%, have their plans set to automatically renew.

```
In [26]: df_trans.is_auto_renew.mean()
```

```
Out[26]: 0.7853025382789347
```

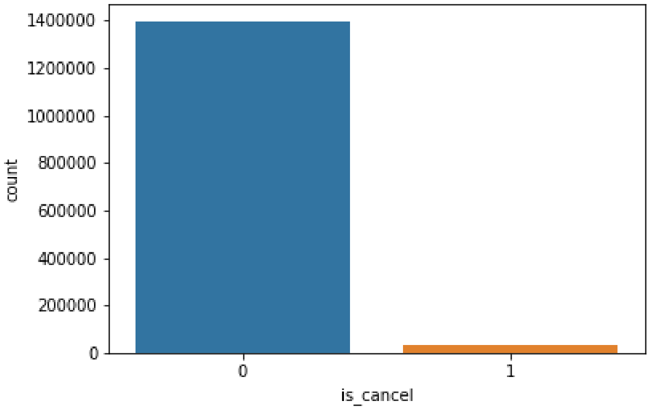
```
In [9]: sns.countplot(x='is_auto_renew',data=df_trans)

        /home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
        l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
          stat_data = remove_na(group_data)

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x7f70e78ed6d8>
```



Only a small proportion of users in the data set canceled their subscriptions. The fraction that cancelled is:

```
In [52]: len(df_trans[df_trans.is_cancel==1])/len(df_trans)

Out[52]: 0.02455120827332323
```

```
In [11]: sns.countplot(x='is_cancel',data=df_trans)

         /home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
         l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
           stat_data = remove_na(group_data)

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f70e7873128>
```

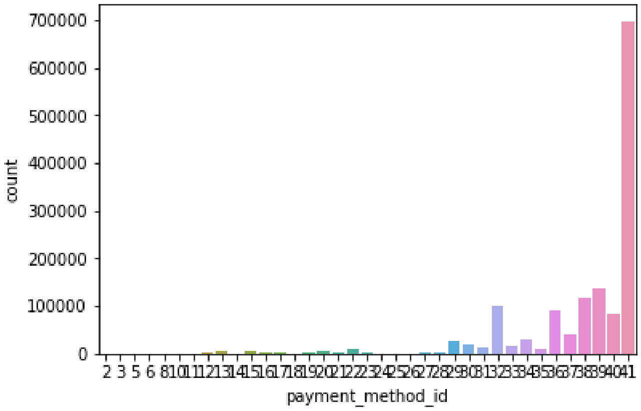


The most popular payment method is 41. Otherwise, payment methods greater than 30 are the most popular.

```
In [53]: sns.countplot(x='payment_method_id',data=df_trans)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[53]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1e6754fe48>
```

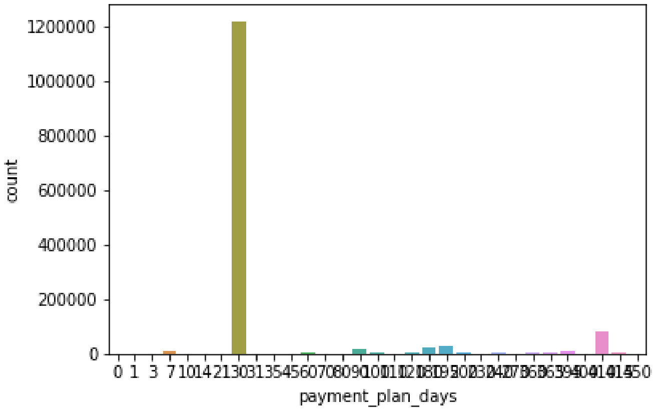


The most popular payment plan duration is 30 days, followed by multiples of months (60 days, 90 days) and 1 week.

```
In [57]: sns.countplot(x='payment_plan_days',data=df_trans)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

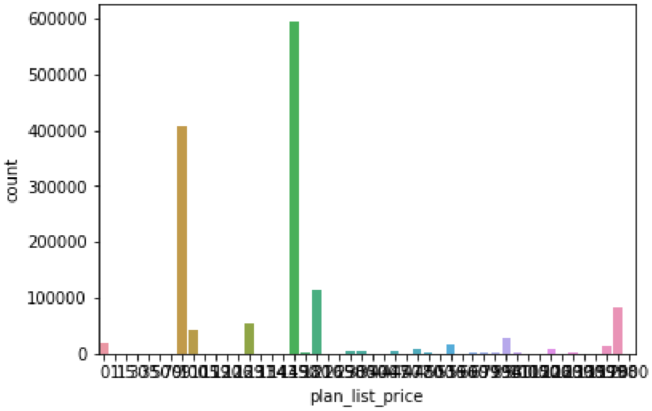
```
Out[57]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1df239acc0>
```



```
In [59]: sns.countplot(x='plan_list_price',data=df_trans)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

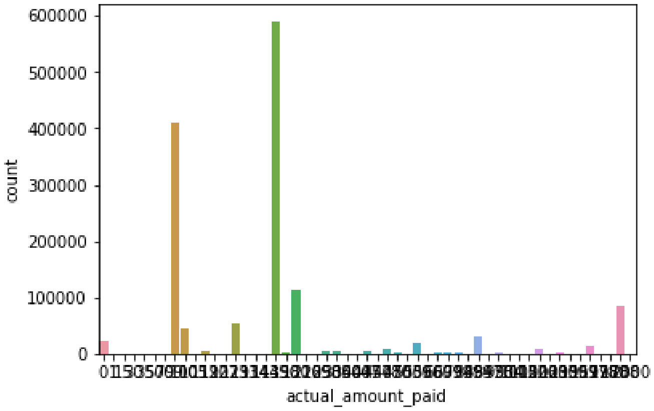
```
Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1df2201208>
```



```
In [60]: sns.countplot(x='actual_amount_paid',data=df_trans)
```

```
/home/rebecca/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorica
l.py:1460: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  stat_data = remove_na(group_data)
```

```
Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1df2085b00>
```



About 1% of the users were not

```
In [63]: (df_trans['plan_list_price']==df_trans['actual_amount_paid']).value_counts()
```

```
Out[63]: True     1419106
         False      11903
         dtype: int64
```

The mean number of unique songs listened to daily is about 29.

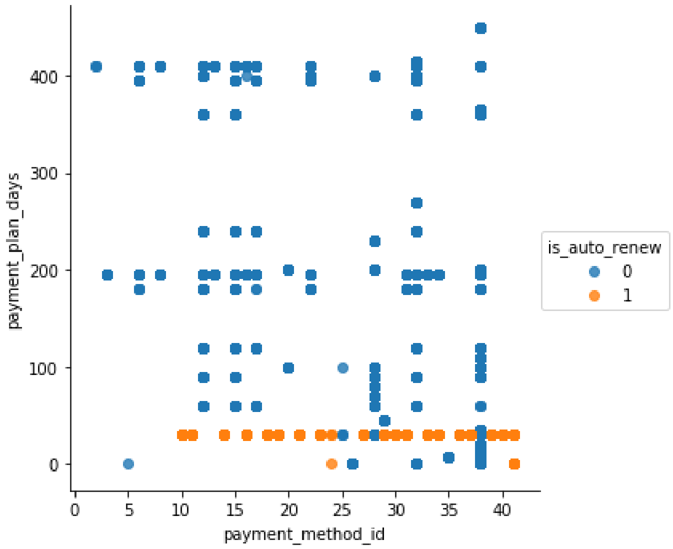
```
In [32]: df_logs.num_unq.mean()
```

```
Out[32]: 29.036145516162382
```

All of the auto_renew plans have only 2 lengths (payment_plan_days).

```
In [22]: sns.lmplot(x="payment_method_id", y="payment_plan_days", hue="is_auto_renew", data=df_trans
         ,fit_reg=False)
```

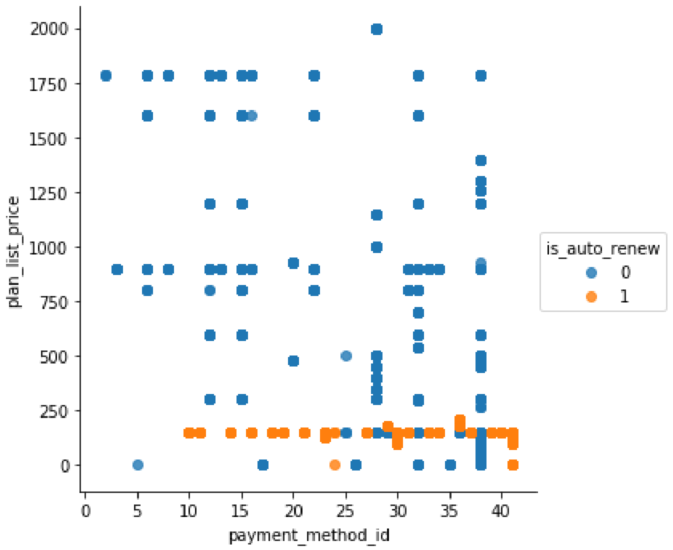
```
Out[22]: <seaborn.axisgrid.FacetGrid at 0x7f70c6f5bd30>
```



These two auto_renew plans have a few different prices.

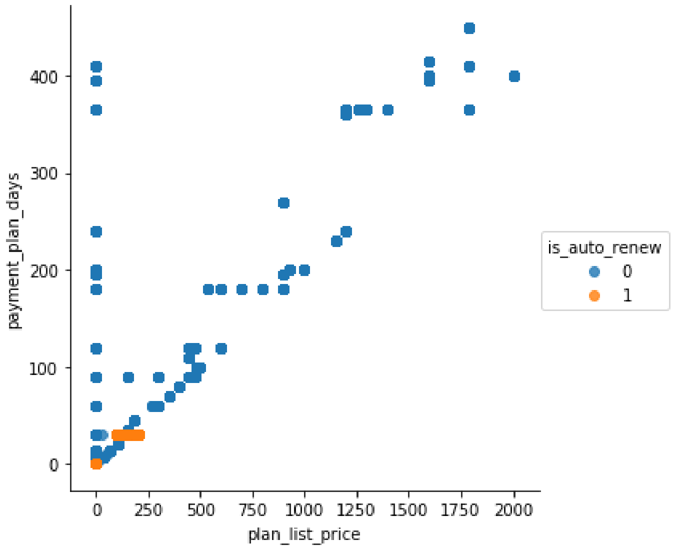
```
In [28]: sns.lmplot(x="payment_method_id", y="plan_list_price", hue="is_auto_renew", data=df_trans,f
         it_reg=False)
```

```
Out[28]: <seaborn.axisgrid.FacetGrid at 0x7f70c6ec49e8>
```



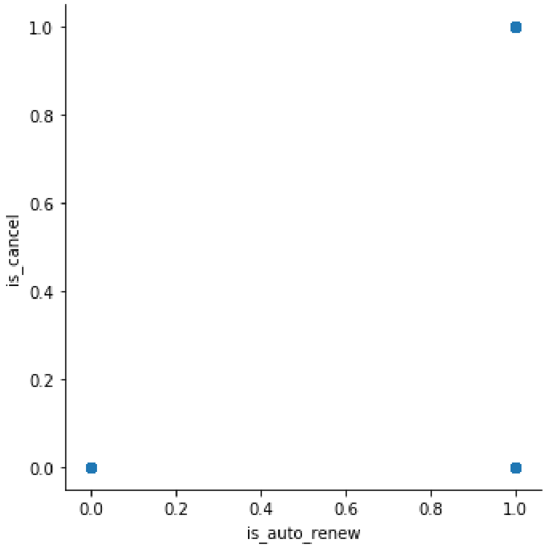
From the scatterplot of payment_plan_days vs plan_list_price it looks like there is a max plan price for each plan duration.

```
In [27]: sns.lmplot(x="plan_list_price", y="payment_plan_days", hue="is_auto_renew", data=df_trans,f
         it_reg=False)

Out[27]: <seaborn.axisgrid.FacetGrid at 0x7f70c2c166d8>
```



Auto renew is never false when is_cancel is true.

```
In [30]: sns.lmplot(x="is_auto_renew", y="is_cancel", data=df_trans,fit_reg=False)

Out[30]: <seaborn.axisgrid.FacetGrid at 0x7f70c2baecc0>
```



In [19]:

```
df_trans.groupby('msno')['is_auto_renew'].apply(lambda x: x.mode())
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-19-b09305af209d> in <module>()
----> 1 df_trans.groupby('msno')['is_auto_renew'].apply(lambda x: x.mode())

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/groupby.py in appl
y(self, func, *args, **kwargs)
    719         # ignore SettingWithCopy here in case the user mutates
    720         with option_context('mode.chained_assignment', None):
--> 721             return self._python_apply_general(f)
    722 
    723     def _python_apply_general(self, f):

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/groupby.py in _pyt
hon_apply_general(self, f)
    723     def _python_apply_general(self, f):
    724         keys, values, mutated = self.grouper.apply(f, self._selected_obj,
--> 725                                                    self.axis)
    726 
    727         return self._wrap_applied_output(

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/groupby.py in appl
y(self, f, data, axis)
   1878             # group might be modified
   1879             group_axes = _get_axes(group)
-> 1880             res = f(group)
   1881             if not _is_indexed_like(res, group_axes):
   1882                 mutated = True

<ipython-input-19-b09305af209d> in <lambda>(x)
----> 1 df_trans.groupby('msno')['is_auto_renew'].apply(lambda x: x.mode())

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/series.py in mode
(self)
   1277         """
   1278         # TODO: Add option for bins like value_counts()
-> 1279         return algorithms.mode(self)
   1280 
   1281     @Appender(base._shared_docs['unique'] % _shared_doc_kwargs)

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/algorithms.py in m
ode(values)
    675 
    676     result = _reconstruct_data(result, original.dtype, original)
--> 677     return Series(result)
    678 
    679 

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/series.py in __ini
t__(self, data, index, dtype, name, copy, fastpath)
    262             else:
    263                 data = _sanitize_array(data, index, dtype, copy,
--> 264                                        raise_cast_failure=True)
    265 
    266                 data = SingleBlockManager(data, index, fastpath=True)

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/pandas/core/series.py in _sani
tize_array(data, index, dtype, copy, raise_cast_failure)
   3132 
   3133     # GH #846
-> 3134     if isinstance(data, (np.ndarray, Index, Series)):
   3135 
   3136         if dtype is not None:

KeyboardInterrupt: 
```

```
In [20]: sns.barplot(x="is_auto_renew", y="is_churn", data=df_trans)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-20-85fcf67b30d4> in <module>()
----> 1 sns.barplot(x="is_auto_renew", y="is_churn", data=df_trans)

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorical.py in barp
lot(x, y, hue, data, order, hue_order, estimator, ci, n_boot, units, orient, color, palett
e, saturation, errcolor, errwidth, capsize, dodge, ax, **kwargs)
   2938                           estimator, ci, n_boot, units,
   2939                           orient, color, palette, saturation,
-> 2940                           errcolor, errwidth, capsize, dodge)
   2941 
   2942     if ax is None:

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorical.py in __in
it__(self, x, y, hue, data, order, hue_order, estimator, ci, n_boot, units, orient, color,
 palette, saturation, errcolor, errwidth, capsize, dodge)
   1584         """Initialize the plotter."""
   1585         self.establish_variables(x, y, hue, data, orient,
-> 1586                                  order, hue_order, units)
   1587         self.establish_colors(color, palette, saturation)
   1588         self.estimate_statistic(estimator, ci, n_boot)

~/anaconda3/envs/my_projects_env/lib/python3.6/site-packages/seaborn/categorical.py in esta
blish_variables(self, x, y, hue, data, orient, order, hue_order, units)
    149                 if isinstance(input, string_types):
    150                     err = "Could not interpret input '{}'".format(input)
--> 151                     raise ValueError(err)
    152 
    153             # Figure out the plotting orientation

ValueError: Could not interpret input 'is_churn'
```

```
In [33]: df = df_trans.copy()
         df['is_churn'] = df_train['is_churn']
```

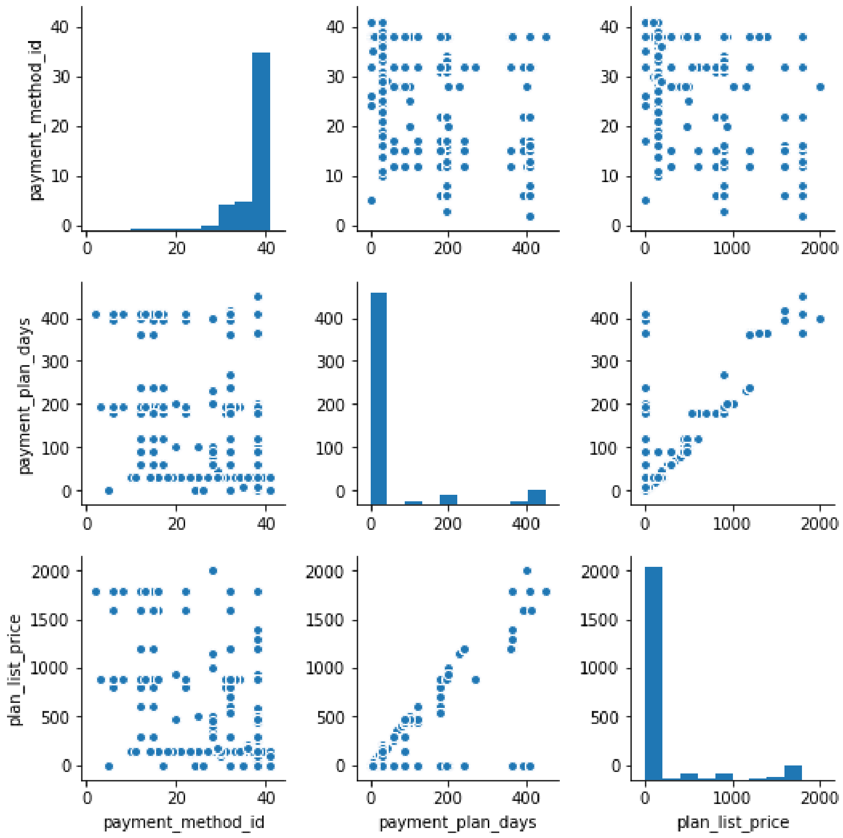
```
In [15]: len(df_members[df_members['bd']>100])

Out[15]: 5377
```

gender is missing for some data points. Check to see whether all msno exist in all tables.

```
In [42]: df1 = df_members.copy()
         df1['is_churn'] = df_train['is_churn']
```

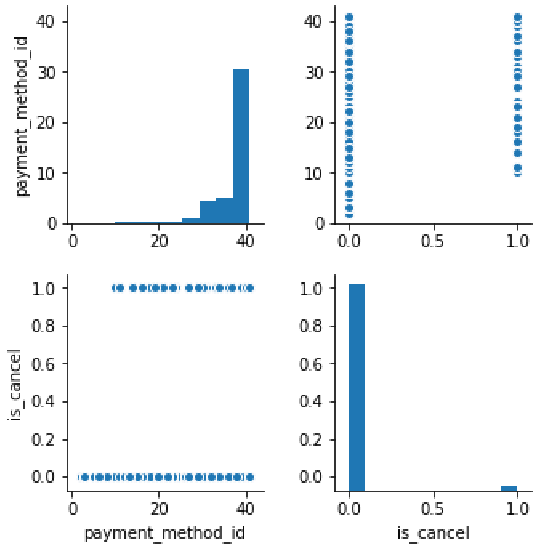
```
In [49]: sns.pairplot(df[['payment_method_id', 'payment_plan_days', 'plan_list_price']],diag_kind='hist')
         #for ax in g.axes.flat:
         #    plt.setp(ax.get_xticklabels(), rotation=45)
```

```
Out[49]: <seaborn.axisgrid.PairGrid at 0x7f3289dd7cc0>
```



```
In [51]: sns.pairplot(df[['payment_method_id', 'is_cancel']],diag_kind='hist')
```

```
Out[51]: <seaborn.axisgrid.PairGrid at 0x7f328134ce48>
```



```
In [ ]: transactions.hist(column=['actual_amount_paid','plan_list_price'],figsize=(16, 5),bins=50)
```

```
In [ ]: gr.plot.line(y=['num_unq'],figsize=(16, 5))
```