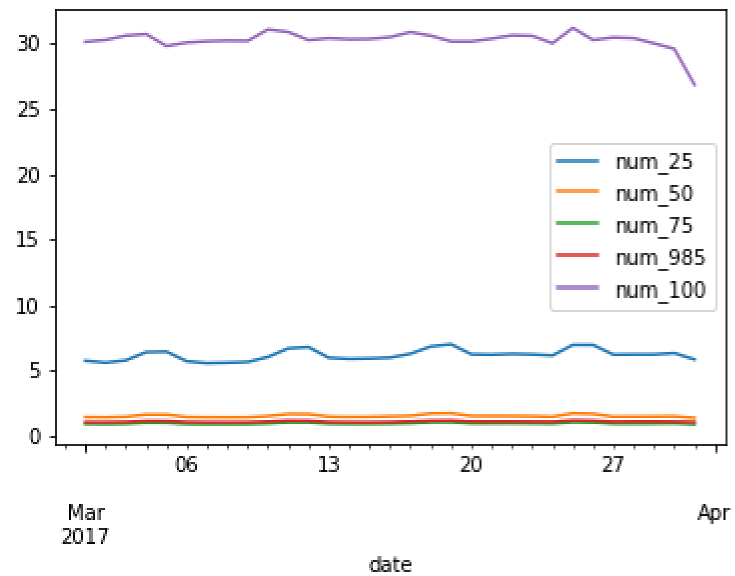
In [32]:
```
num_listened_plot.plot()
```

Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x7f73eb169908>



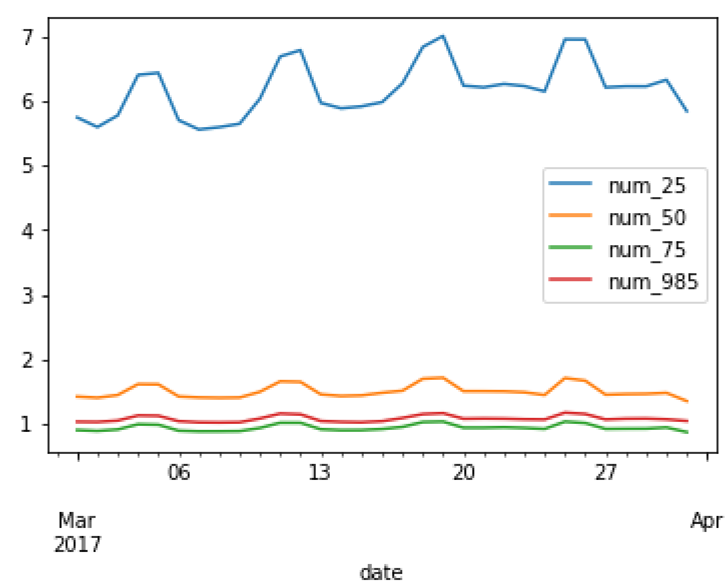
After songs played up to 100% of the song length, on average, users listen daily to more songs played less than 25% of the song length, than songs played at other lengths.

This could be because users are interested in trying out songs that they are not familiar with. The daily average number of songs played between 25% and 50%, between 50% and 75%, and between 75% and 98.5% of the length are very similar.

In [33]:
```
num_listened_plot_2 = df_logs.groupby('date')['num_25','num_50','num_
75','num_985'].agg('mean')
```

In [34]:
```
num_listened_plot_2.plot()
```

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7f741f12b2b0>

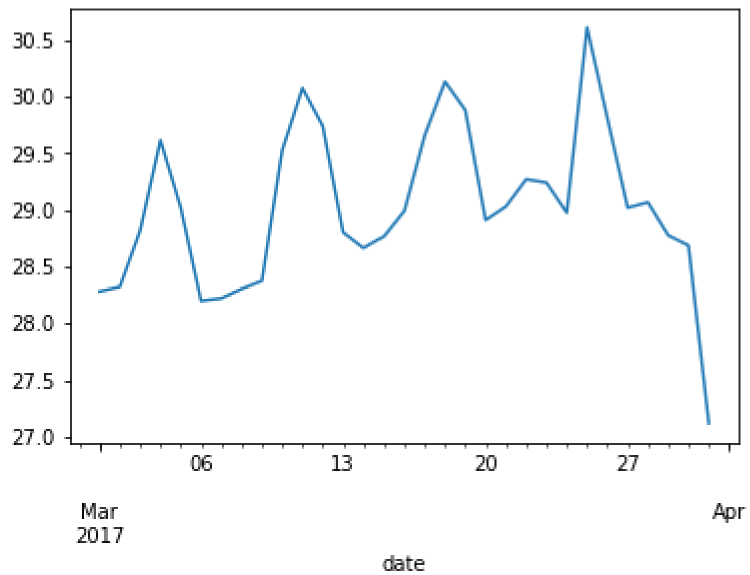


The daily average number of unique songs listened to fluctuates throughout the month, peaking on the weekends.

In [14]:
```
num_unq_plot = df_logs.groupby('date')['num_unq'].agg('mean')
```

In [27]:
```
num_unq_plot.plot()
```

Out[27]: `<matplotlib.axes._subplots.AxesSubplot at 0x7f73bce777f0>`



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

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

Out[32]: 29.036145516162382

Inferential Statistics

In [46]:
```
# def one_hot(column,df):
#     df_dummies = pd.get_dummies(df[column])
#     del df_dummies[df_dummies.columns[-1]]
#     df_new = pd.concat([df, df_dummies], axis=1)
#     del df_new[column]
#     return df_new
```

all_data = one_hot('payment_method_id',all_data) all_data = one_hot('city',all_data) all_data = one_hot('gender',all_data) all_data = one_hot('registered_via',all_data) all_data = one_hot('registration_init_time',all_data) all_data = one_hot('year',all_data)

In [58]:
```
all_data_onehot = pd.get_dummies(all_data, prefix=None, prefix_sep='_', dummy_na=False, columns=['payment_method_id','city','gender','registered_via','registration_init_time'])
```

Whether or not a user churned is most strongly correlated to payment_plan_days, plan_list_price, actual_amount_paid, payment_method_id_32, is_auto_renew, and is_cancel. These features have correlations between 0.47 and 0.31 with is_churn.

In [59]:
```
corr = all_data_onehot.corr()
c = corr.abs()
s = c.unstack()
```

In [60]:
```
s['is_churn'].sort_values(ascending=False)
```

```
Out[60]: is_churn                        1.000000
         payment_plan_days               0.473736
         plan_list_price                 0.455707
         actual_amount_paid              0.450579
         payment_method_id_32            0.384989
         is_auto_renew                   0.349667
         is_cancel                       0.313537
         payment_method_id_15            0.193123
         payment_method_id_38            0.159720
         payment_method_id_41            0.155421
         registered_via_7                0.147419
         payment_method_id_20            0.128193
         transaction_date_count          0.105310
         city_1                          0.101843
         payment_method_id_22            0.096334
         registered_via_4                0.083453
         payment_method_id_17            0.078696
         payment_method_id_13            0.077872
         registered_via_3                0.072665
         registered_via_9                0.065339
         bd                              0.062260
         payment_method_id_12            0.062150
         gender_female                   0.057437
         payment_method_id_35            0.054600
         gender_male                     0.054083
         payment_method_id_37            0.038075
         city_13                         0.036822
         city_5                          0.034938
         payment_method_id_34            0.034064
         payment_method_id_26            0.032196
                                           ...
         city_3                          0.008955
         city_11                         0.008257
         payment_method_id_23            0.008207
         payment_method_id_27            0.008109
         registration_init_time_2015     0.007750
         registration_init_time_2005     0.007235
         city_18                         0.006577
         registration_init_time_2012     0.006363
         payment_method_id_19            0.005347
         registration_init_time_2014     0.005195
         registration_init_time_2008     0.005125
         payment_method_id_29            0.004922
         registration_init_time_2017     0.004311
         payment_method_id_14            0.003915
         payment_method_id_18            0.003912
         registration_init_time_2006     0.003140
         city_7                          0.002967
         registration_init_time_2004     0.002550
         payment_method_id_21            0.002353
         payment_method_id_11            0.001838
         registration_init_time_2013     0.001534
         city_19                         0.001474
         registered_via_13               0.001116
         city_16                         0.001042
         payment_method_id_10            0.000939
         payment_method_id_30            0.000937
```

```
registration_init_time_2007    0.000687
city_17                        0.000654
registration_init_time_2011    0.000567
city_20                        0.000095
Length: 90, dtype: float64
```

A t-test is used to see whether there is a statistical difference in the proportion of female vs male churners. At a confidence level of 0.01, we reject the null hypothesis that the proportion of female and male churners is the same.

In [62]:
```
from scipy.stats import ttest_ind, f_oneway
ttest_ind(all_data['is_churn'][all_data.gender=='female'],all_data['is_churn'][all_data.gender=='male'])
```

Out[62]:
```
Ttest_indResult(statistic=3.0423516232159784, pvalue=0.0023475520227502554)
```

A t-test is used to see whether there is a statistical difference in the proportion of churners who have canceled and those who have not canceled. At a confidence level of 0.01, we reject the null hypothesis that the proportion of churners who have canceled is the same as the proportion of churners who have not canceled.

In [16]:
```
ttest_ind(all_data['is_churn'][all_data.is_cancel==1],all_data['is_churn'][all_data.is_cancel==0])
```

Out[16]:
```
Ttest_indResult(statistic=281.28261945889841, pvalue=0.0)
```

A one way ANOVA is used to see whether there is a statistical difference in the proportion of churners who registered in 2017, vs those who registered in 2016 and 2015. At a confidence level of 0.01, we reject the null hypothesis that the proportion of churners who registered in 2017, 2016, and 2015 is the same.

In [20]:
```
f_oneway(all_data['is_churn'][all_data.registration_init_time==2017],all_data['is_churn'][all_data.registration_init_time==2016],all_data['is_churn'][all_data.registration_init_time==2015])
```

Out[20]:
```
F_onewayResult(statistic=43.652813916443328, pvalue=1.1078849825135133e-19)
```

A one way ANOVA is used to see whether there is a statistical difference in the proportion of churners who are in city 1, vs those who are in cities 3 and 4. At a confidence level of 0.01, we reject the null hypothesis that the proportion of churners who are in cities 1, 3, and 4 are the same.

In [26]:
```
f_oneway(all_data['is_churn'][all_data.city==1],all_data['is_churn'][all_data.city==3],all_data['is_churn'][all_data.city==4])
```

Out[26]:
```
F_onewayResult(statistic=1254.7043807999448, pvalue=0.0)
```

In [ ]:
```
#t-test is_discount
```