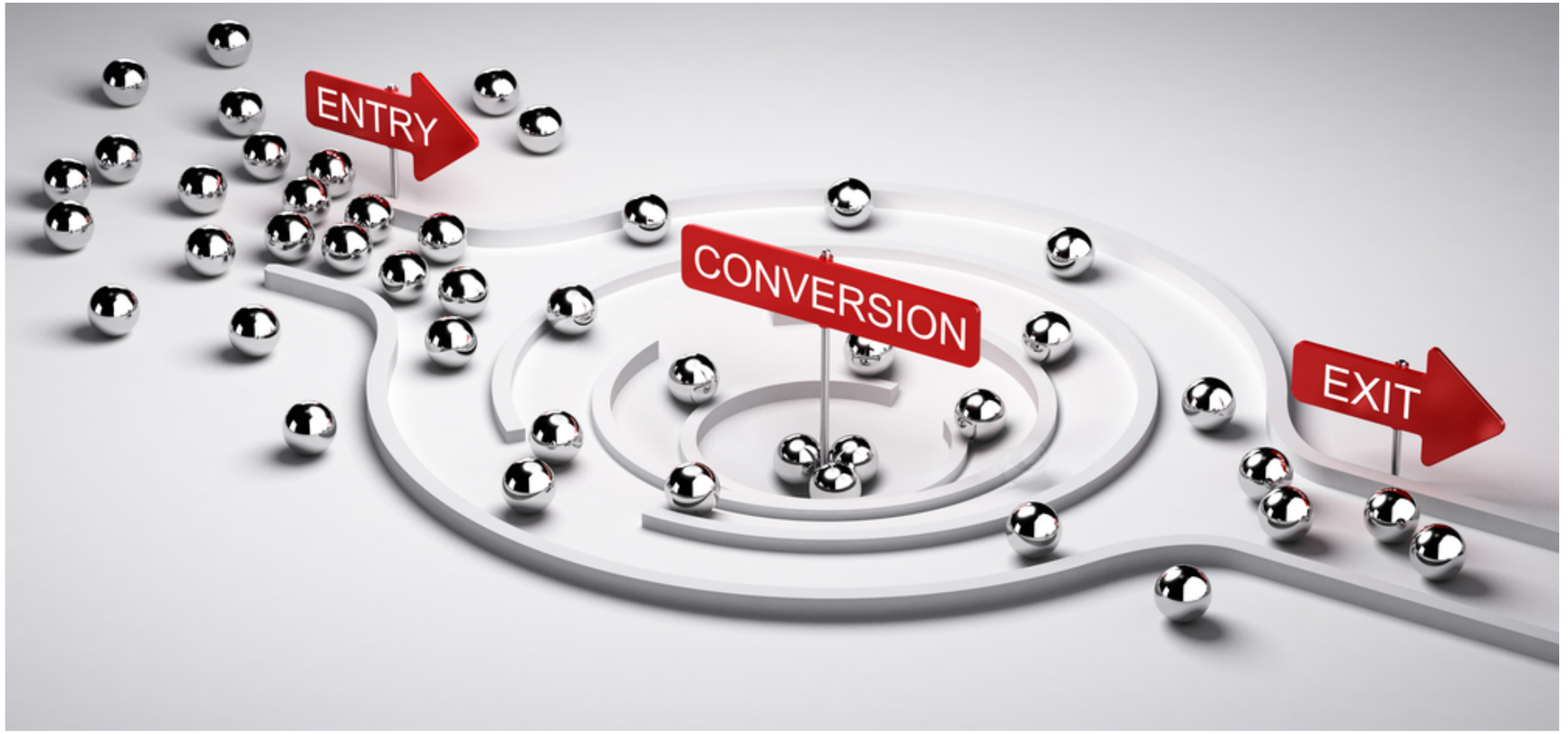
CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Working with time series data in pandas

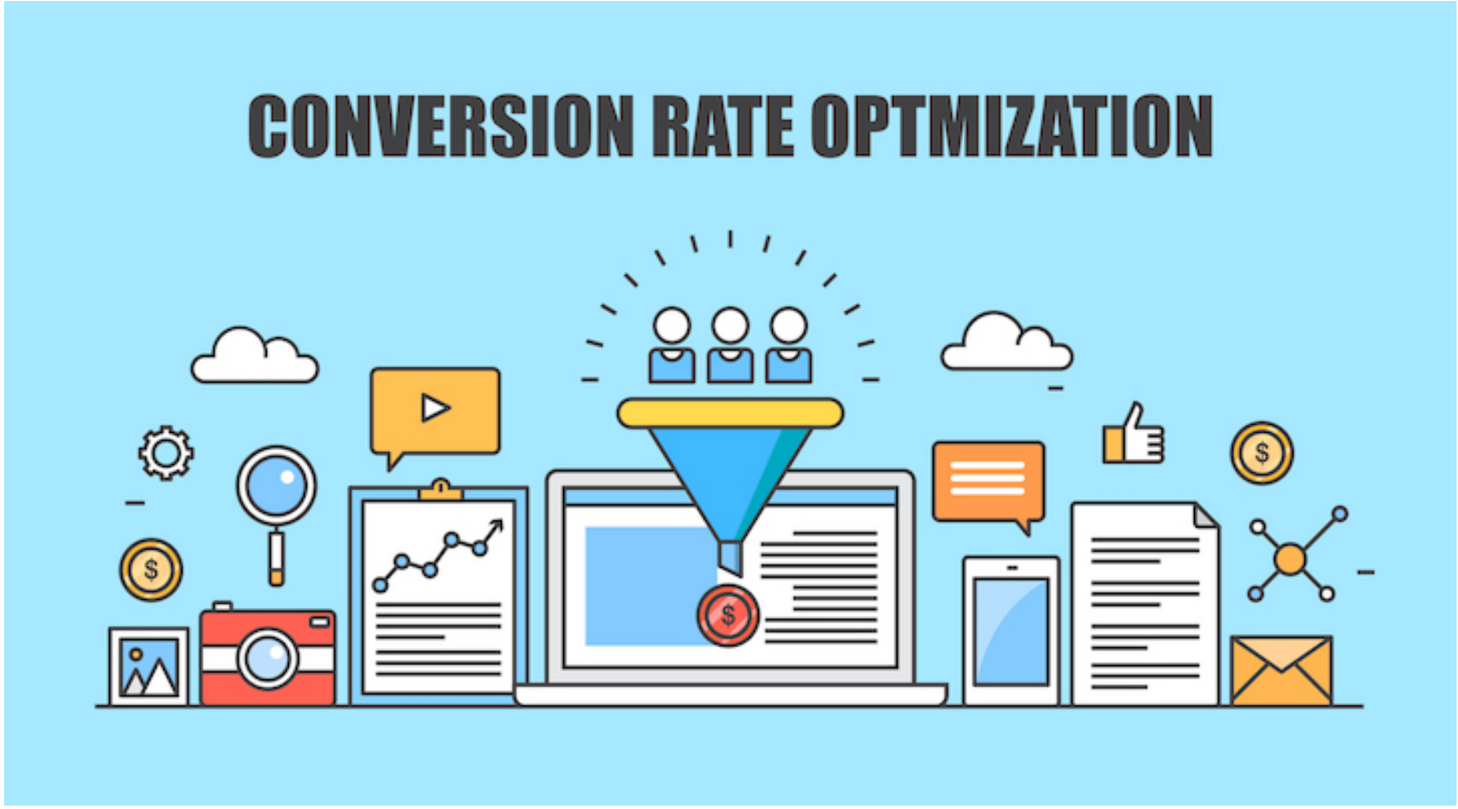
Ryan Grossman
Data Scientist, EDO

Exploratory Data Analysis

Dates & Times



Week Two Conversion Rate



Using the Timedelta Class

```
In [1]: current_date = pd.to_datetime('2018-03-17')

In [2]: max_lapse_date = current_date - timedelta(days=14)

In [3]: conv_sub_data = sub_data_demo[
    sub_data_demo.lapse_date < max_lapse_date]
```

Date Differences

```
In [4]: sub_time = (conv_sub_data.subscription_date
                    - conv_sub_data.lapse_date)

In [5]: conv_sub_data['sub_time'] = sub_time
```

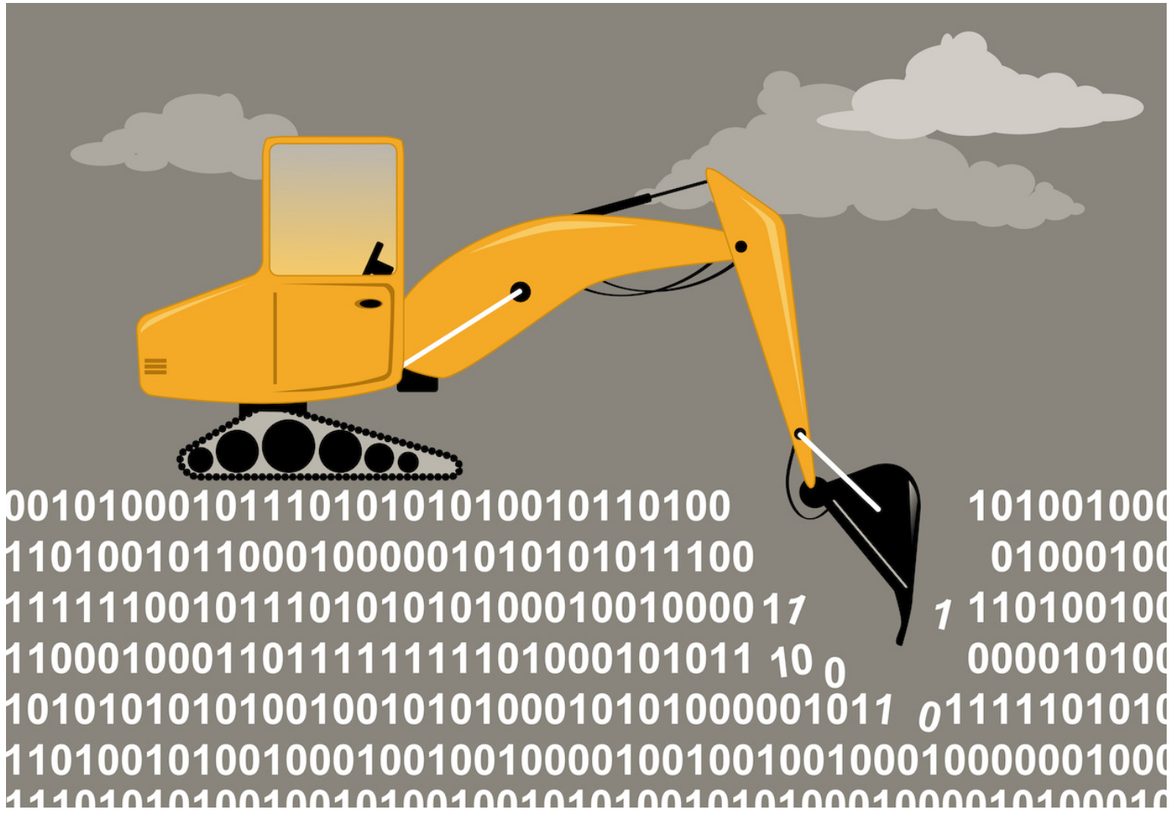
Date Components

```
In [6]: conv_sub_data['sub_time'] = conv_sub_data.sub_time.dt.days
```

Conversion Rate Calculation

```
In [7]: conv_base = conv_sub_data[
    (conv_sub_data.sub_time.notnull()) | (conv_sub_data.sub_time > 7)]

In [8]: total_users = len(conv_base)

In [9] total_users
Out[9]: 2086

In [10]: total_subs = np.where(conv_sub_data.sub_time.notnull()
    & (conv_base.sub_time <= 14), 1, 0)

In [11] total_subs = sum(total_subs)

In [12] total_subs
Out[12] 20

In [13] conversion_rate = total_subs / total_users

In [14] conversion_rate
Out[14]: 0.0095877277085330784
```

Digging Deeper



Parsing Dates - On Import

```
pandas.read_csv(...,
    parse_dates=False,
    infer_datetime_format=False,
    keep_date_col=False,
    date_parser=None,
    dayfirst=False,
    ...)
```

```
In [15]: customer_demographics = pd.read_csv('customer_demographics.csv',
    parse_dates=True,
    infer_datetime_format=True
)

Out [15]:
    uid            reg_date          device gender    country    age
0    54030035.0    2017-06-29        and        M        USA        19
1    72574201.0    2018-03-05        iOS        F        TUR        22
2    64187558.0    2016-02-07        iOS        M        USA        16
3    92513925.0    2017-05-25        and        M        BRA        41
4    99231338.0    2017-03-26        iOS        M        FRA        59
```

Parsing Dates - Manually

```
pandas.to_datetime(arg, errors='raise', ..., format=None, ...)
```

strftime

1993-01-27 --`"%Y-%m-%d"`

05/13/2017 05:45:37 -- `"%m/%d/%Y %H:%M:%S"`

September 01, 2017 -- `"%B %d, %Y"`

CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Let's practice!

CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Creating time series graphs With matplotlib

Ryan Grossman
Data Scientist, EDO

Conversion Rate Over Time

Monitoring The Impact of Changes

Conversion Rate by Day

```
In [1]: current_date = pd.to_datetime('2018-03-17')

In [2]: max_lapse_date = current_date - timedelta(days=7)

In [3]: conv_sub_data = sub_data_demo[sub_data_demo.lapse_date
                                < max_lapse_date]

In [4]: sub_time = (conv_sub_data.subscription_date -
                            conv_sub_data.lapse_date).dt.days

In [5]: conv_sub_data['sub_time'] = sub_time
```

Conversion Rate by Day

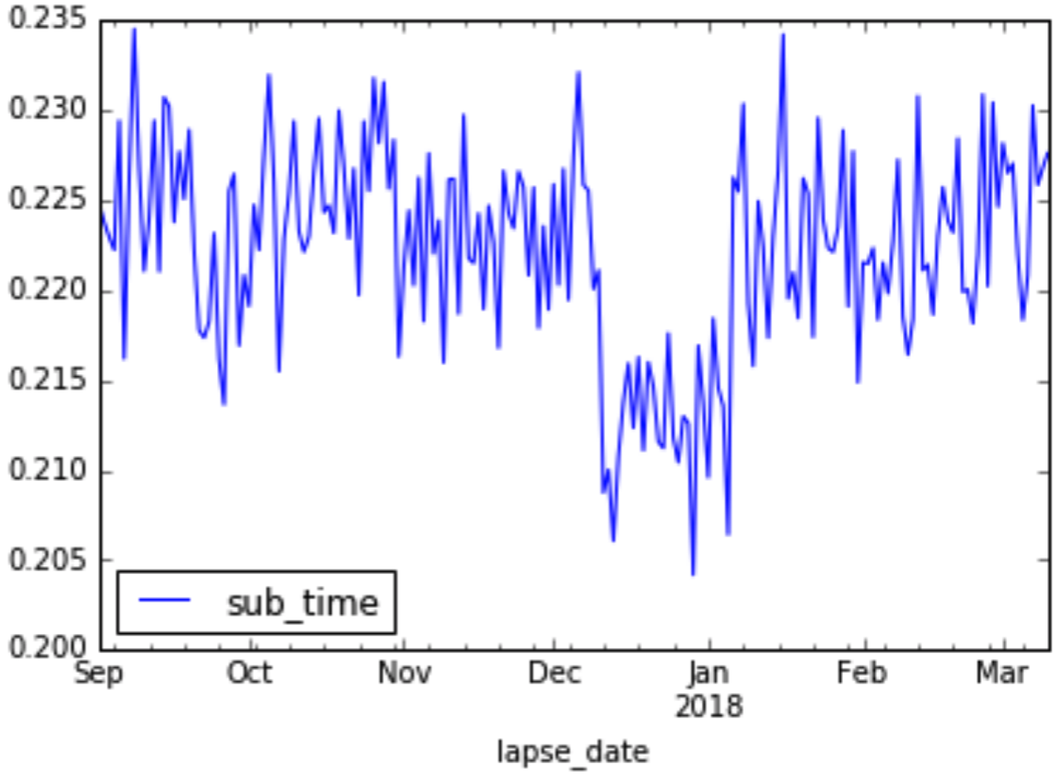
```
In [6]: conversion_data = conv_sub_data.groupby(by=['lapse_date'],
                                 as_index=False)

In [7]: conversion_data = conversion_data.agg({'sub_time': [gc7]})

In [9]: conversion_data.columns = conversion_data.columns.droplevel(
                                       level=1)

In [8]: conversion_data.head()

Out[8]:
    lapse_date    sub_time
0    2017-09-01    0.224775
1    2017-09-02    0.223749
2    2017-09-03    0.222948
3    2017-09-04    0.222222
4    2017-09-05    0.229401
```

Plotting Daily Conversion Rate

```
In [9]: conversion_data.lapse_date =
            pd.to_datetime(conversion_data.lapse_date)

In [10]: conversion_data.plot(x='lapse_date', y='sub_time')
```

Plotting Daily Conversion Rate

```
In [13]: plt.show()
```



Trends in Different Cohorts

```
In [11]: conversion_data.head()

Out[11]:
    lapse_date    country    sub_time
0    2017-09-01     BRA        0.184000
1    2017-09-01     CAN        0.285714
2    2017-09-01     DEU        0.276119
3    2017-09-01     FRA        0.240506
4    2017-09-01     TUR        0.161905
```

.pivot_table()

Pivot Table Method

```
pandas.pivot_table(
    data, values=None, index=None, columns=None,
    aggfunc='mean', fill_value=None, margins=False,
    dropna=True, margins_name='All')
```

.pivot_table()

```
In [12]: reformatted_cntry_data =pd.pivot_table(conversion_data, ...)

In [13]: reformatted_cntry_data =pd.pivot_table(conversion_data,
    values=['sub_time'],...)

In [14]: reformatted_cntry_data =pd.pivot_table(conversion_data,
    values=['sub_time'], columns=['country'],
    ...)

In [15]: reformatted_cntry_data =pd.pivot_table(conversion_data,
        values=['sub_time'],columns=['country'],
        index=['reg_date'],fill_value=0 )
```

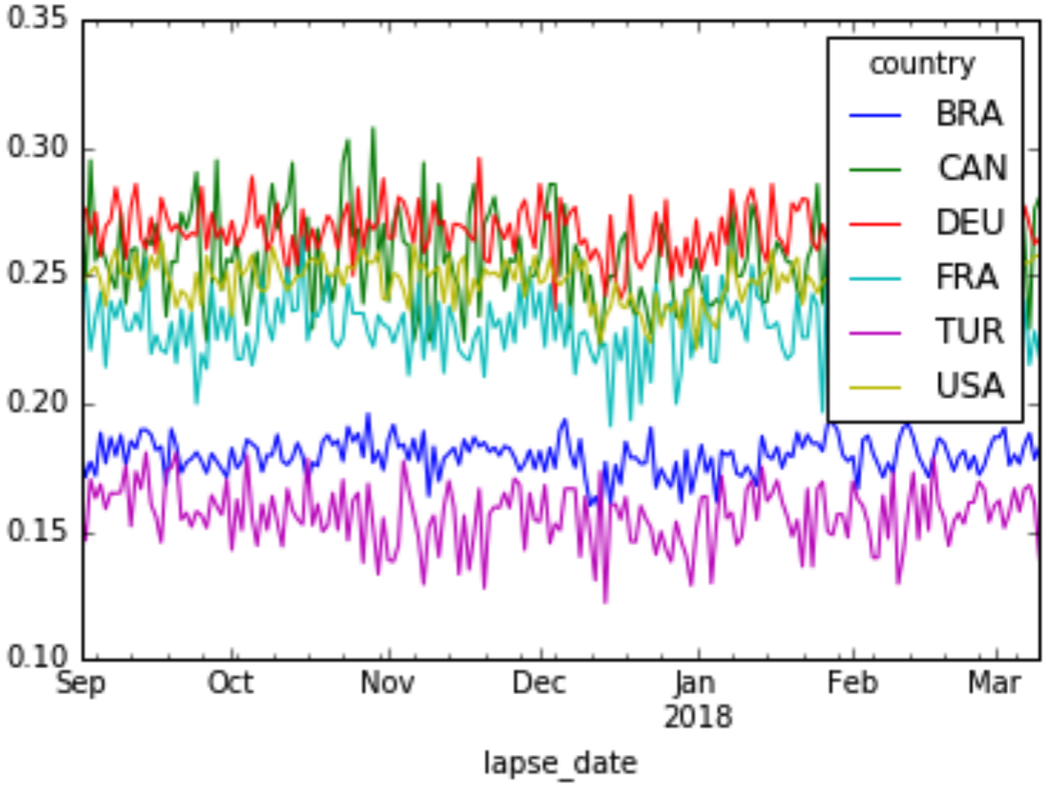
.pivot_table()

```
In [16]: reformatted_cntry_data.columns
    reformatted_cntry_data.columns.droplevel(level=[0])

In [17]: reformatted_cntry_data.reset_index(inplace=True)

In [18]: reformatted_cntry_data.head()

Out[18]:
lapse_date          BRA          CAN          DEU
2017-09-01     0.184000     0.285714     0.276119    ...
2017-09-02     0.171296     0.244444     0.276190    ...
2017-09-03     0.177305     0.295082     0.266055    ...
```

Plotting Trends in Different Cohorts

```
In [19]: reformatted_cntry_data.plot(
     x='reg_date',
     y=['BRA','FRA','DEU','TUR','USA','CAN']
)

In [20]: plt.show()
```

Plotting Trends in Different Cohorts



CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Let's practice!

CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Understanding and visualizing trends in customer data

Ryan Grossman
Data Scientist, EDO

Further Techniques for Uncovering Trends

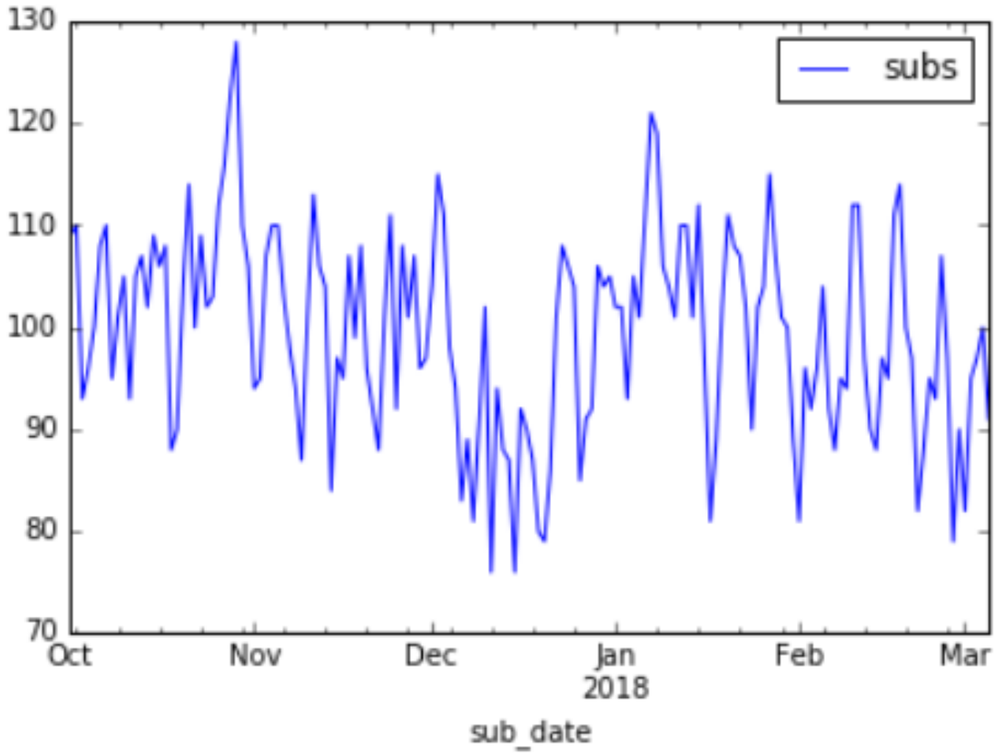
Subscribers Per Day

```
In [1]: usa_subscriptions = pd.read_csv('usa_subscribers.csv',
                                        parse_dates=True,
                                        infer_datetime_format=True)

In [2]: usa_subscriptions['sub_day'] = (usa_subscriptions.sub_date -
                                        usa_subscriptions.lapse_date).dt.days

In [3]: usa_subscriptions = usa_subscriptions[
                                        usa_subscriptions.sub_day <= 7]

In [4]: usa_subscriptions = usa_subscriptions.groupby(
                                        by=['sub_date'], as_index = False)

In [5]: usa_subscriptions = usa_subscriptions.agg({'subs': ['sum']})

In [6]: usa_subscriptions.columns = usa_subscriptions.columns.droplevel(
          level=[1])

In [7]: usa_subscriptions.head()

Out[7]:
    sub_date         subs
0    2016-09-02        37
1    2016-09-03        50
2    2016-09-04        59
```

Subscribers Per Day

```
In [8]: usa_subscriptions.plot(x='sub_date', y='subs')

In [9]: plt.show()
```

Seasonality



Correcting for Seasonality

Trailing Averages

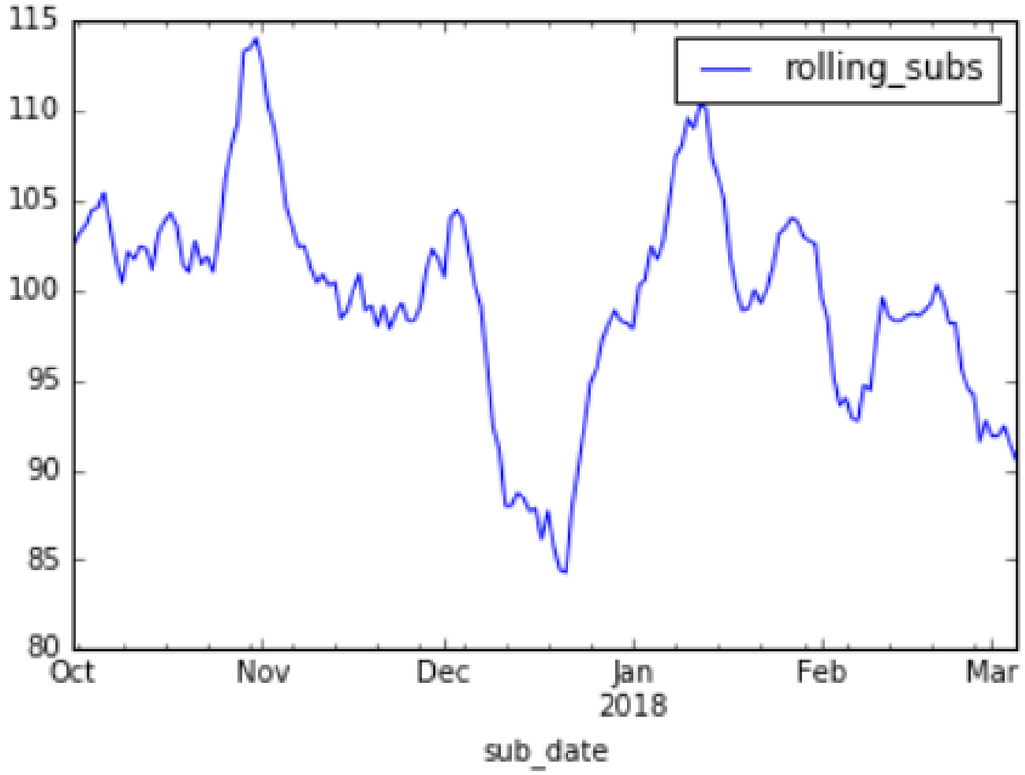
Calculating Trailing Averages

```
In [10]: rolling_subs =  usa_subscriptions.subs.rolling(...)

In [10]: rolling_subs =  usa_subscriptions.subs.rolling(window=7, ...)

In [10]: rolling_subs = usa_subscriptions.subs.rolling(
           window=7, center=False)
```

Calculating Trailing Averages

```
In [11]: rolling_subs = rolling_subs.mean()

In [12]: usa_subscriptions['rolling_subs'] = rolling_subs

In [13]: usa_subscriptions.tail()

Out[13]:
sub_date    subs        rolling_subs
2018-03-14  89          94.714286
2018-03-15  96          95.428571
2018-03-16  102         96.142857
2018-03-17  102         96.142857
2018-03-18  115         98.714286
```

Smoothed Data

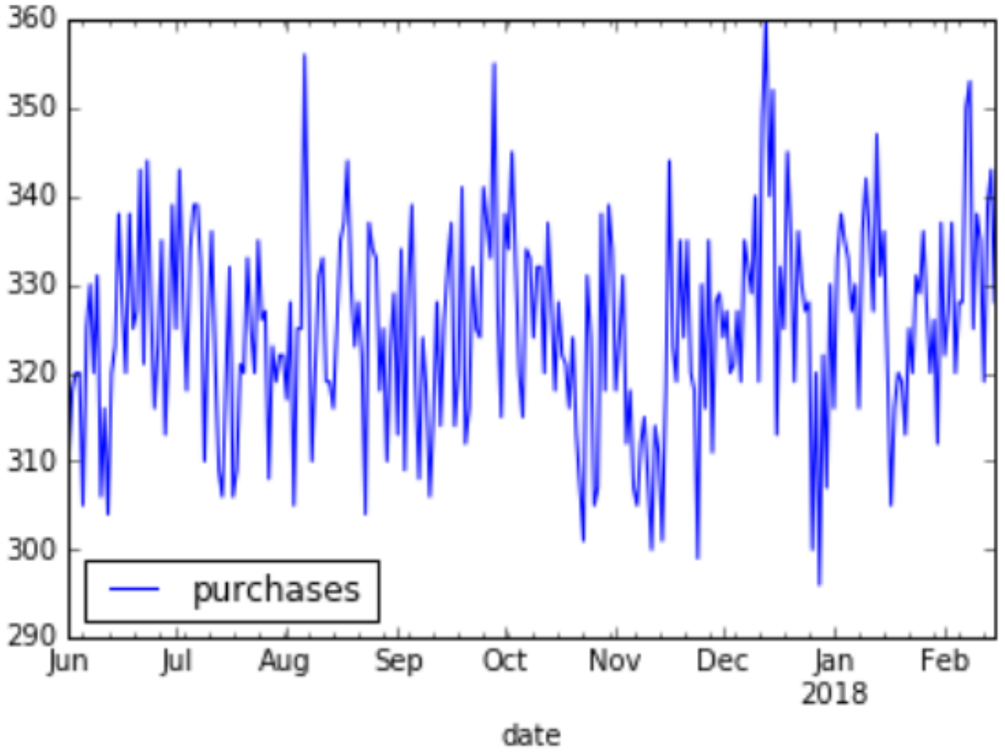


Noisy Data

Noisy Data

```
In [14]: high_sku_purchases = pd.read_csv('high_sku_purchases.csv',
                                          parse_dates=True,
                                          infer_datetime_format=True)

In [15]: high_sku_purchases.plot(x='date', y='purchases')

In [16]: plt.show()
```

Exponential Moving Average

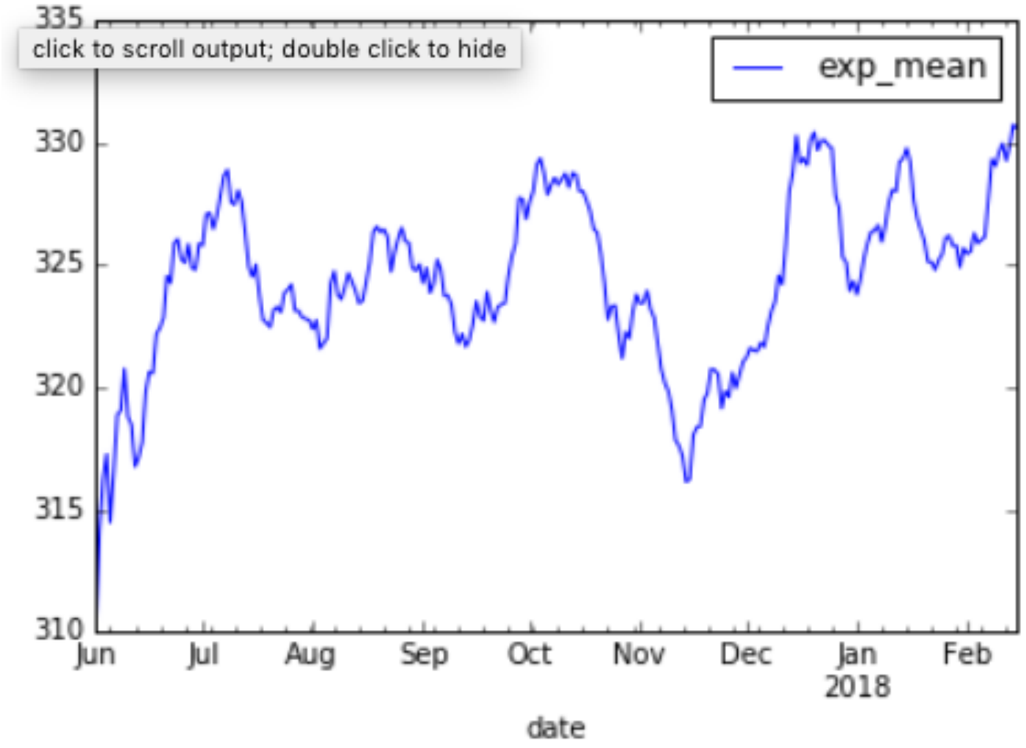


Calculating an Exponential Moving Average

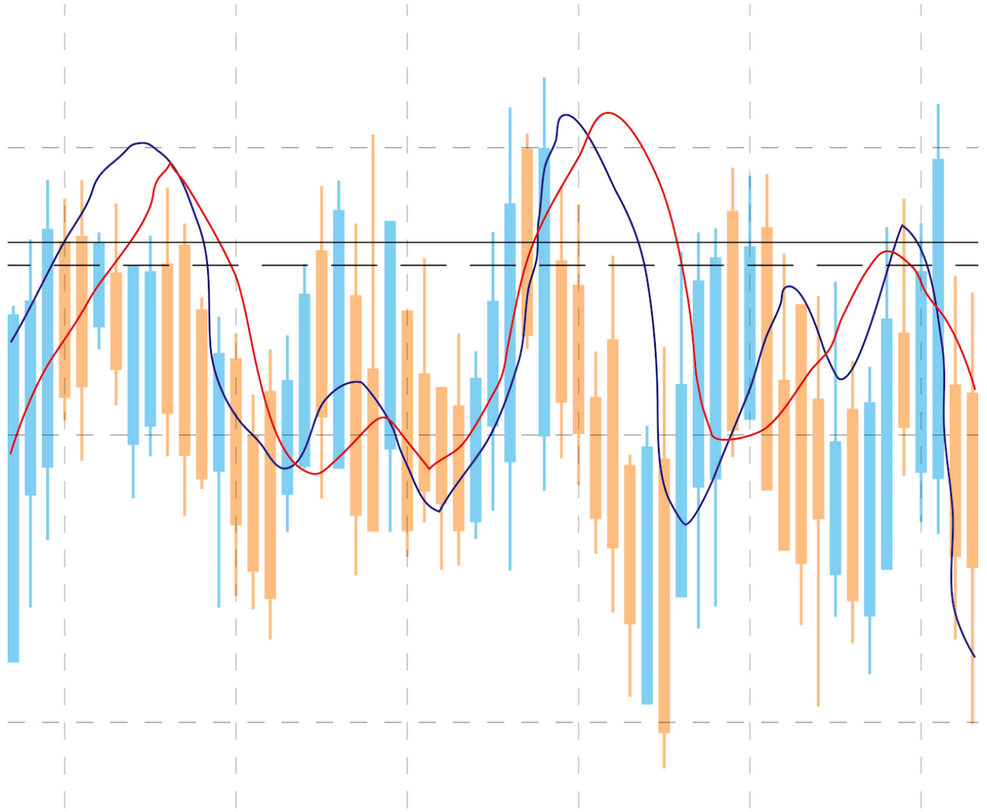
```
In [17]: exp_mean = high_sku_purchases.purchases.ewm(span=30)

In [18]: exp_mean = exp_mean.mean()

In [19]: high_sku_purchases['exp_mean'] = exp_mean
```

Calculating an Exponential Moving Average

```
In [20]: high_sku_purchases.plot(x='date', y='exp_mean')
```



Data Smoothing Techniques



CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Let's practice!

CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Exploratory data analysis with time series data

Ryan Grossman
Data Scientist, EDO

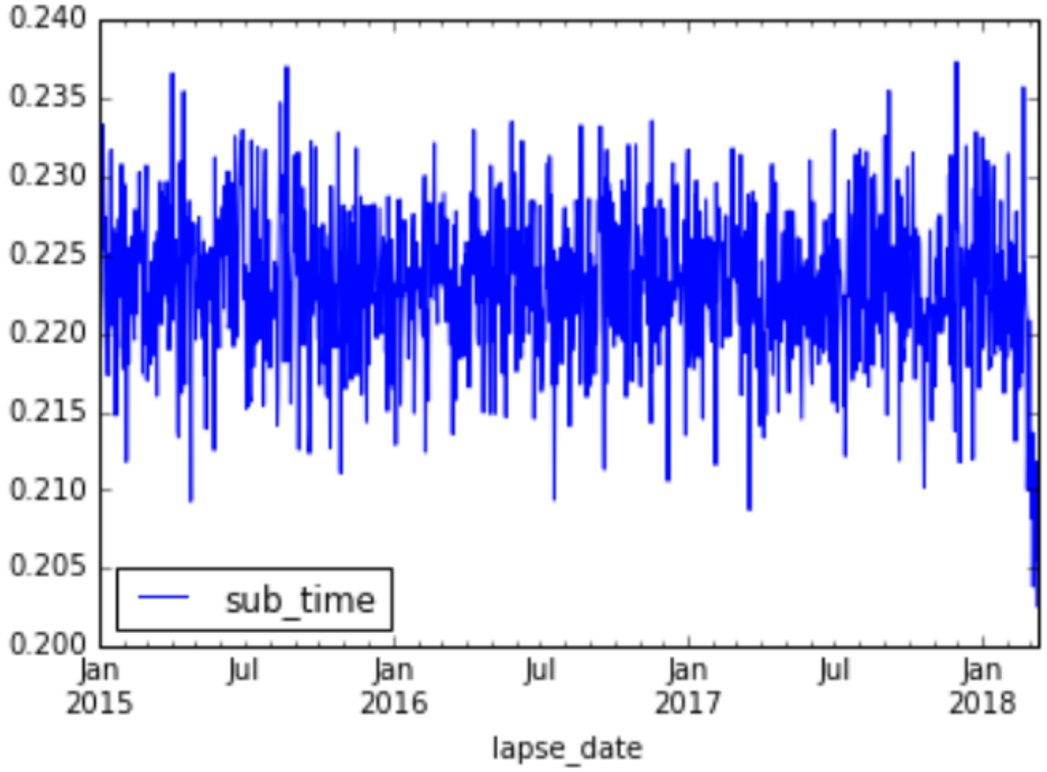
Exploratory Analysis

Drop in New User Retention

```
In [1]: current_date = pd.to_datetime('2018-03-17')
In [2]: max_lapse_date = current_date - timedelta(days=7)
In [3]: conv_sub_data = sub_data_demo[sub_data_demo.lapse_date
                                      <= max_lapse_date]

In [4]: sub_time = (conv_sub_data.subscription_date -
                    conv_sub_data.lapse_date).dt.days
In [6]: conv_sub_data['sub_time'] = sub_time

In [7]: conversion_data = conv_sub_data.groupby(by=['lapse_date'],
                                  as_index=False)
In [8]: conversion_data = conversion_data.agg({'sub_time': [gc7]})
In [9]: conversion_data.columns = conversion_data.columns.droplevel(level=1)
```

```
In [10]: conversion_data.plot()
In [11]: plt.show()
```

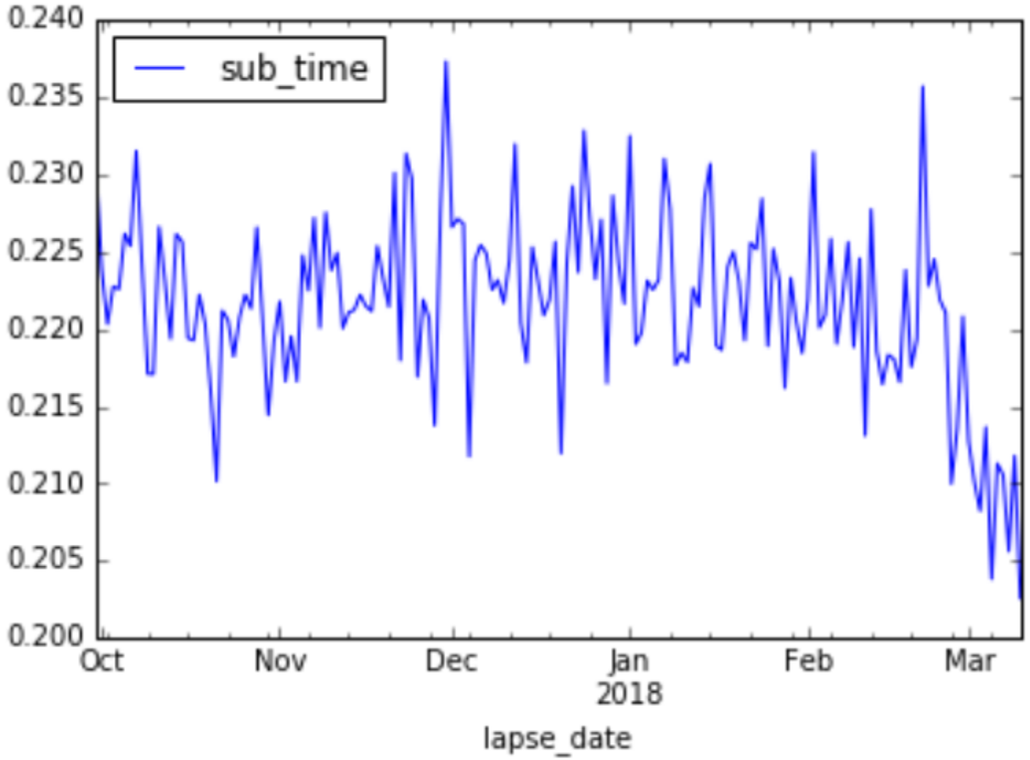
Limiting our View



Limiting our View

```
In [12]: current_date = pd.to_datetime('2018-03-17')

In [13]: start_date = current_date - timedelta(days=(6*28))

In [14]: conv_filter = ((conversion_data.lapse_date >= start_date) &
                        (conversion_data.lapse_date <= current_date))

In [15]: conversion_data_filt = conversion_data[conv_filter]

In [16]: conversion_data_filt.plot(x='lapse_date', y='sub_time')

In [17]: plt.show()
```

Uncovering the Dip



Segmenting our Graph

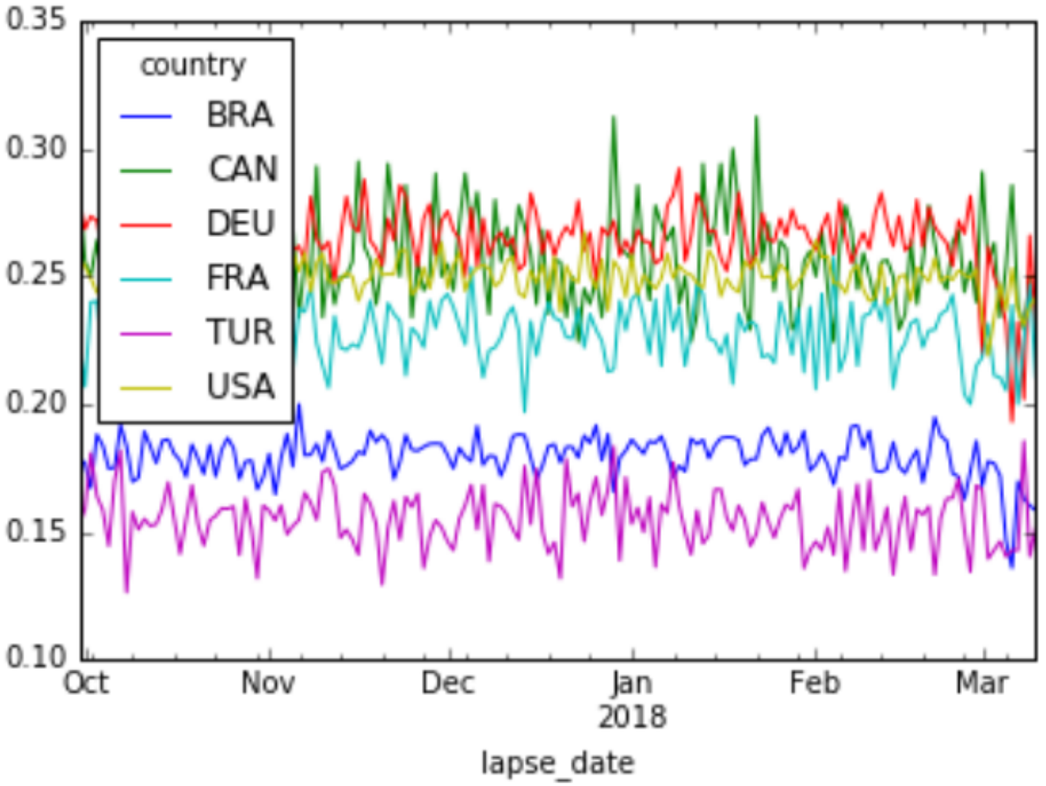
Splitting by Country & Device

```
In [18]: conv_filter = ((conv_sub_data.lapse_date >= start_date) &
                        (conv_sub_data.lapse_date <= current_date))

In [19]: conv_data = conv_sub_data[conv_filter]

In [20]: conv_data_cntry = conv_data.groupby(by=['lapse_date', 'country'],
                                as_index=False)

In [20]: conv_data_cntry = conv_data_cntry.agg({'sub_time': [gc7]})

In [21]: conv_data_cntry.columns = conv_data_cntry.columns.droplevel(level=1)

In [22]: conv_data_cntry = pd.pivot_table(conv_data_cntry,
                                values=['sub_time'],
                                columns=['country'],
                                index=['lapse_date'],fill_value=0 )

In [23]: conv_data_cntry.columns = conv_data_cntry.columns.droplevel(
                                    level=0)

In [24]: conv_data_cntry.reset_index(inplace=True)

In [25]: conv_data_cntry.plot(x=['lapse_date'],
            y=['BRA', 'CAN', 'DEU', 'FRA', 'TUR', 'USA'])

In [26]: plt.show()
```

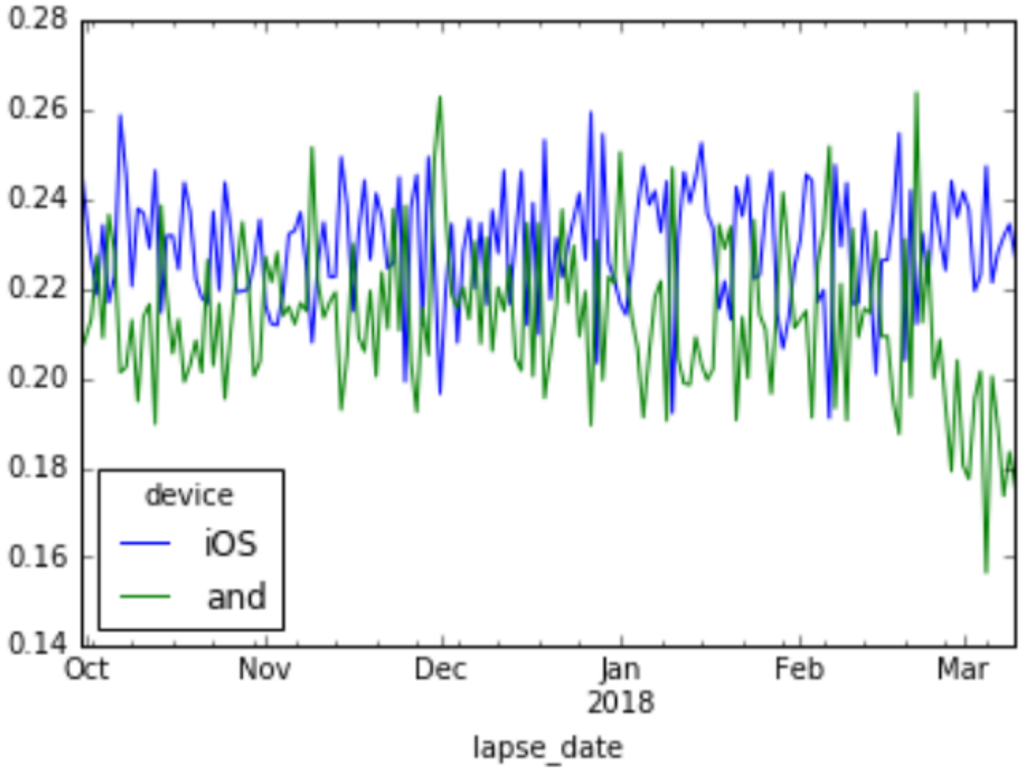
Splitting by Country & Device

```
In [27]: conv_filter = ((conv_sub_data.lapse_date >= start_date) &
                        (conv_sub_data.lapse_date <= current_date))

In [28]: conv_data = conv_sub_data[conv_filter]

In [29]: conv_data_dev = conv_data.groupby(by=['lapse_date',
                        'device'], as_index=False)

In [30]: conv_data_dev = conv_data_dev.agg({'sub_time': [gc7]})

In [31]: conv_data_dev.columns = conv_data_dev.columns.droplevel(level=1)

In [32]: conv_data_dev = pd.pivot_table(conv_data_dev, values=['sub_time'],
                        columns=['device'],
                        index=['lapse_date'],fill_value=0 )

In [33]: conv_data_dev.columns = conv_data_dev.columns.droplevel(level=[0])

In [34]: conv_data_dev.reset_index(inplace=True)

In [35]: conv_data_dev.plot(x=['lapse_date'], y=['iOS', 'and'])

In [36]: plt.show()
```

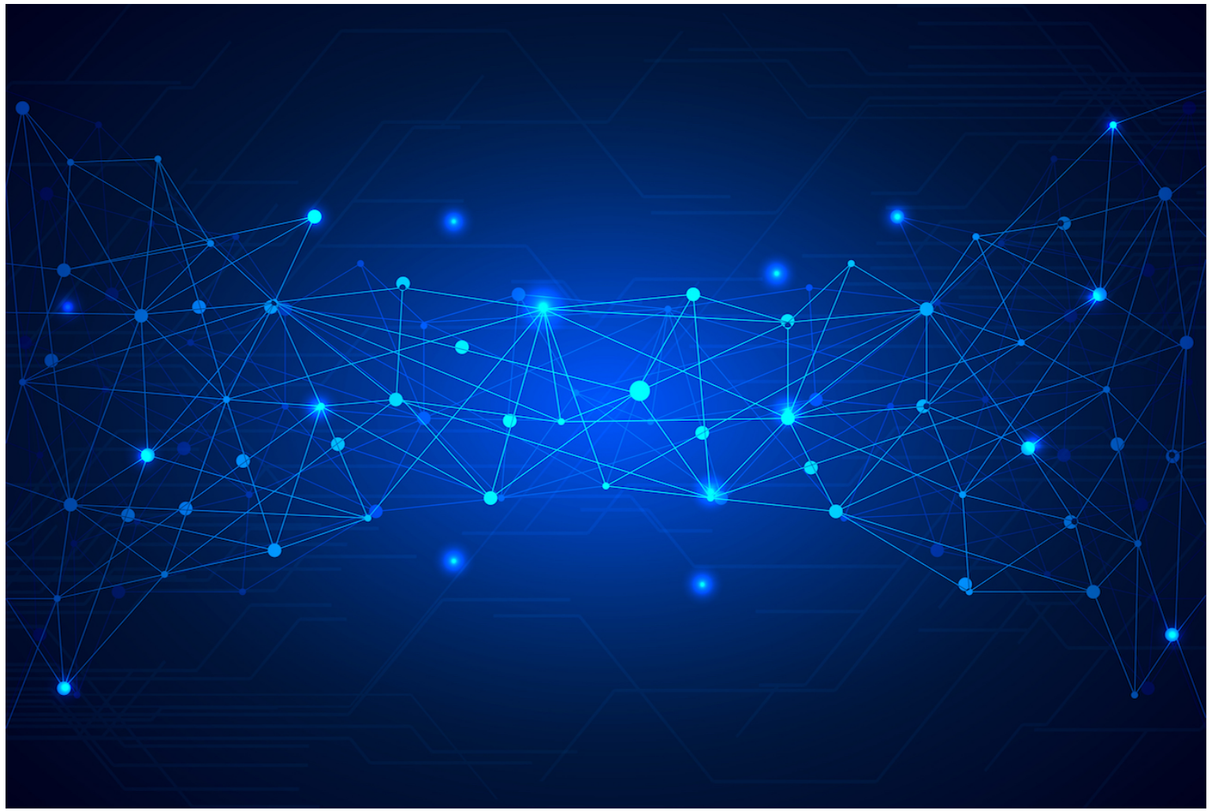
Breaking out by Country



Breaking Out by Device



Adding Annotations



Annotation Datasets

```
In [37]: events = pd.read_csv('events.csv')

In [38]: events.head()

Out[38]:
Date             Event
2018-01-01         NYD
2017-01-01         NYD
2016-01-01         NYD
2015-01-01         NYD
2014-01-01         NYD
```

```
In [39]: releases = pd.read_csv('releases.csv')

In [40]: releases.head()

Out[40]:
    Date            Event
2018-03-14       iOS Release
2018-03-03       Android Release
2018-01-13       iOS Release
2018-01-15       Android Release
2017-11-03       Android Release
```

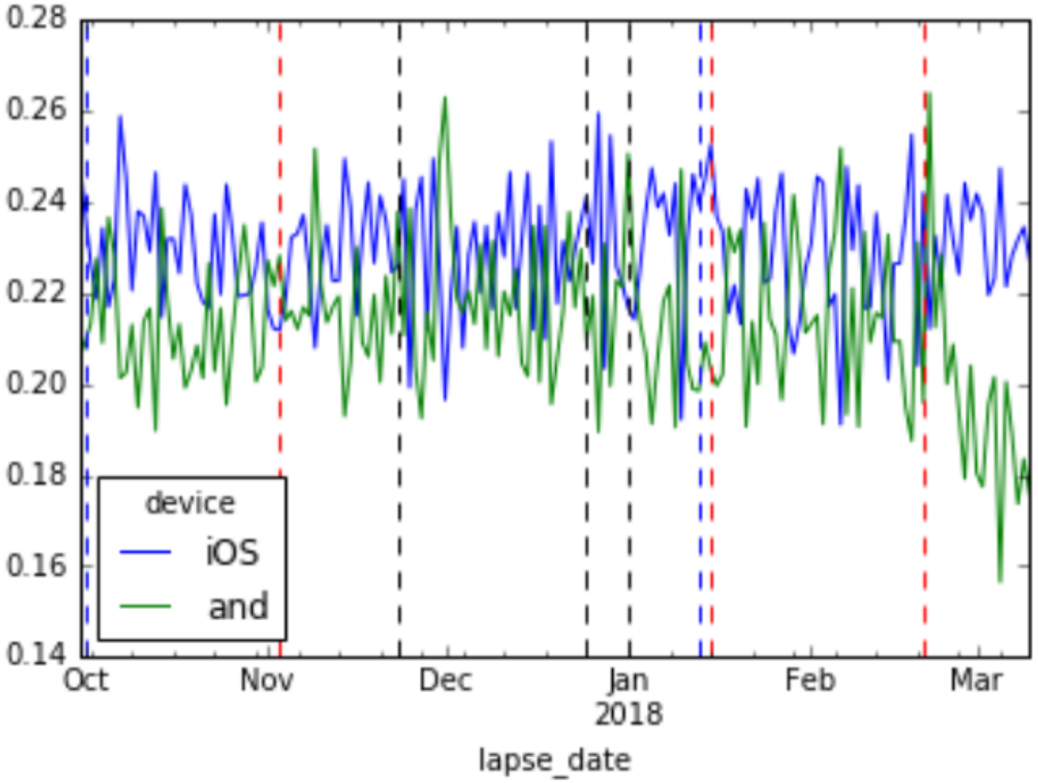
Plotting Annotations

```
In [41]: conv_data_dev.plot(x=['lapse_date'], y=['iOS', 'and'])

In [42]: events.Date = pd.to_datetime(events.Date)

In [43]: for row in events.iterrows():
    tmp = row[1]
    plt.axvline(x=tmp.Date, color='k', linestyle='--')
```

Plotting Annotations

```
In [44]: releases.Date = pd.to_datetime(releases.Date)

In [45]: for row in releases.iterrows():
    tmp = row[1]
    if tmp.Event == 'iOS Release':
        plt.axvline(x=tmp.Date, color='b', linestyle='--')
    else:
        plt.axvline(x=tmp.Date, color='r', linestyle='--')

In [46]: plt.show()
```

Our Final Plot



Exploratory Analysis

CUSTOMER ANALYTICS & A/B TESTING IN PYTHON

Let's practice!