codecademy

Calculating Churn Rates

Learn SQL from Scratch

Fred Triest

Sep 3, 2018 - Oct 29, 2018

Table of Contents

1.0 Get familiar with the company

Get familiar with the company

-Codeflix is storing subscription data by four columns in the database:

id (INTEGER)

subscription_start (TEXT)

subscription_end (TEXT)

segment (INTEGER)

-The id is the primary key of the subscriptions table

-Subscriptions have a start and end date.

-If there is a NULL value (0) for the customer's subscription_end date, then we know the customer hasn't canceled yet.

-There are two segments. Segment 30 and 87.

```sql
--1.0 SELECT all columns from subscriptions table
--and LIMIT query to 100 rows
SELECT *
  FROM subscriptions
 LIMIT 100;
--SELECT and GROUP BY the segment column for concrete
--confirmation on all segments
SELECT segment
  FROM subscriptions
 GROUP BY 1;
```

id	subscription_start	subscription_end	segment
12	2016-12-01	2017-02-07	87
13	2016-12-01	0	30
14	2016-12-01	2017-03-07	30

segment
30
87

1.1 Get familiar with the company – CONT'D

How many months has the company been operating?

-The first subscription start date is 2016-12-01 and the latest subscription start date is 2017-03-30.

-Codeflix may have been operating before 2016-12-01 and might not have been storing subscription data, but from the perspective of the database, Codeflix has been operating for four months.

```sql
--1.1 and 1.2 Use MIN and MAX aggregate functions to retrieve
--the first and latest date values from subscription_start
--column from the subscriptions table
SELECT MIN(subscription_start) AS first_subscription_start_date,
       MAX(subscription_start) AS latest_subscription_start_date
  FROM subscriptions;
```

first_subscription_start_date	lastest_subscription_start_date
2016-12-01	2017-03-30

1.2 Get familiar with the company – CONT'D

Which months do you have enough information to calculate a churn rate?

- In order to calculate subscriber churn rate per month, we divide the number of cancellations within that month by the total subscriber count for each month.
- A **Cancellation** is anyone that was an active subscriber before the start of that month and has and end date within that month.
- **Total Subscribers** is anyone that was an active subscriber before the start of the month and their subscription end date is greater than or equal to the start of the month or they have no subscription end date.

$$\frac{\textbf{Cancellations}}{\textbf{Total Subscribers}}$$

- Since Codeflix requires a minimum subscription length of 31 days and the first ever start date of a subscription is 2016-12-01, there were technically no active subscribers for Dec 2016. This means we can only calculate subscriber churn for January 2017, February 2017, and March 2017.

```sql
--1.1 and 1.2 Use MIN and MAX aggregate functions to retrieve
--the first and latest date values from subscription_start
--column from the subscriptions table
SELECT MIN(subscription_start) AS first_subscription_start_date,
       MAX(subscription_start) AS latest_subscription_start_date
  FROM subscriptions;
```

first_subscription_start_date	lastest_subscription_start_date
2016-12-01	2017-03-30

1.3 Get familiar with the company – CONT'D

What segments of users exist?

-There are two types of segments in the subscriptions table. Segment 30 and Segment 87.

-These are the two distinct channels the Marketing team at Codeflix have acquired the subscribers from.

- Out of the 2,000 subscriptions, 50% are Segment 30 and 50% are Segment 87.

```sql
--1.3 SELECT and GROUP BY the segment column for concrete
--confirmation on all segments
SELECT segment
  FROM subscriptions
 GROUP BY 1;
--SELECT the DISTINCT COUNT of subscribers by id
--and GROUP BY segment to get user count by segment
SELECT DISTINCT COUNT(id) AS number_of_subscribers,
       segment
  FROM subscriptions
 GROUP BY 2;
```

segment
30
87

number_of_subscribers	segment
1000	30
1000	87

2.0 What is the overall churn trend since the company started?

What is the overall churn trend since the company started?

- When disregarding segmentation, the overall churn trend in January is 16%, February is 18%, and March ramps up to 27%.

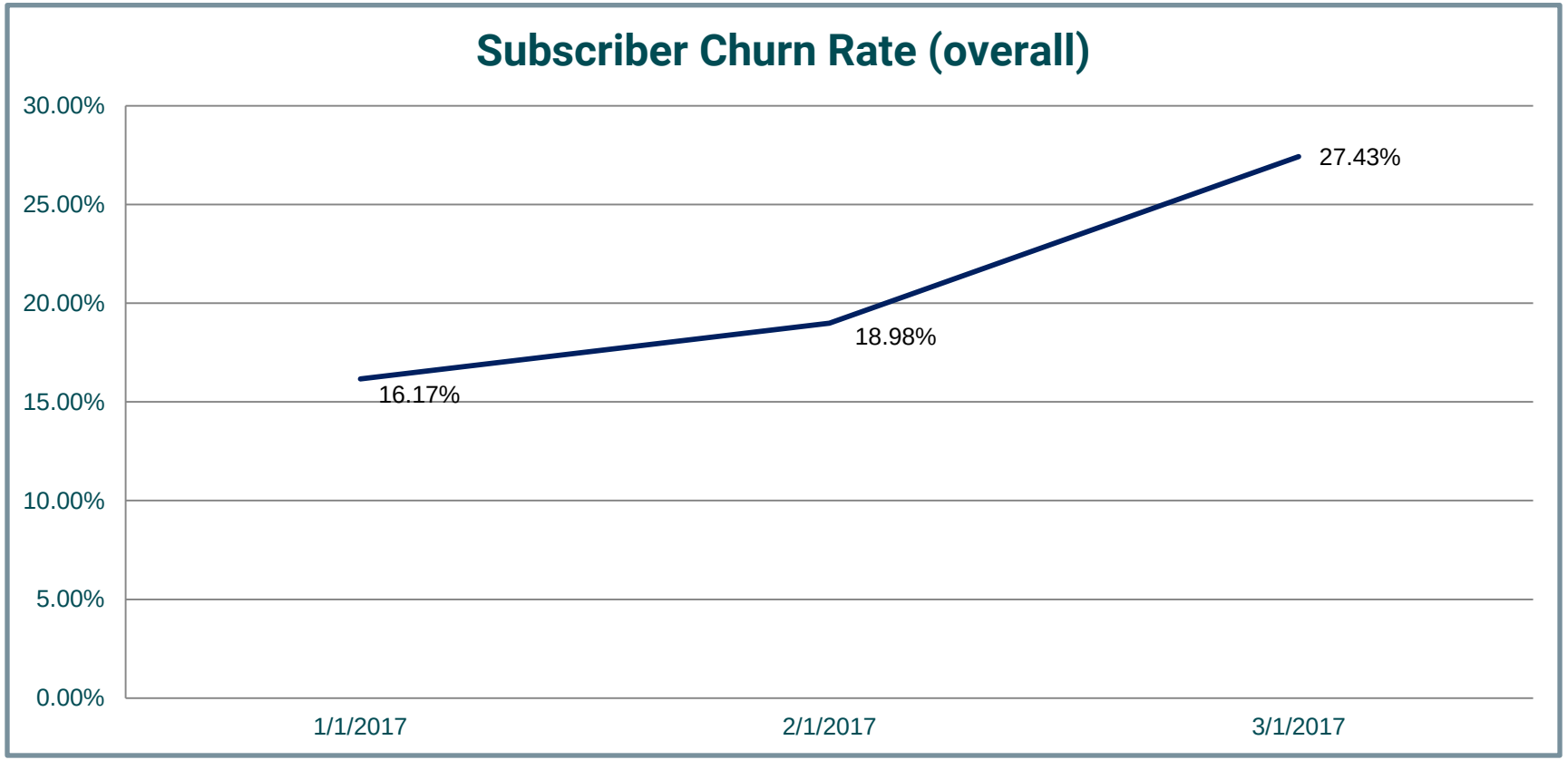
- Codeflix really needs to focus on decreasing subscriber churn at the current rate.

- There may also be seasonal fluctuations so decisions based on 3 months of churn trend analysis in first months of year should be carefully considered, but not ignored.

```sql
--2.0 Create temporary table using WITH and UNION for Jan, Feb,
--and Mar 2017
WITH months AS
(SELECT '2017-01-01' AS first_day,
        '2017-01-31' AS last_day
  UNION
 SELECT
        '2017-02-01' AS first_day,
        '2017-02-28' AS last_day
  UNION
 SELECT
        '2017-03-01' AS first_day,
        '2017-03-31' AS last_day
),
--Create temporary table cross_join to CROSS JOIN with
--subscriptions table
cross_join AS
(SELECT subscriptions.*,
        months.*
   FROM subscriptions
        CROSS JOIN months
),
--Create temporary status table to add 1's and 0's
--in order to SUM them in status_aggregate temporary table.
--Notice that we did not include segment criteria since we want overall churn.
status AS
(SELECT id,
        first_day AS month,
        CASE
          WHEN (subscription start < first day) AND
            (
            subscription_end > first_day OR
            subscription_end IS NULL
            ) THEN 1
          ELSE 0
        END AS is_active,
--Included is_canceled CASE statements to status temporary table
--and closing temporary status table with FROM cross_join
        CASE
          WHEN (subscription_end BETWEEN first_day AND last_day) THEN 1
          ELSE 0
        END AS is_canceled
   FROM cross_join
),
--Create temporary status_aggregate table to SUM
--active and canceled subscriptions as columns and GROUP BY month
status_aggregate AS
(SELECT month,
        SUM(is_active) AS sum_active,
        SUM(is_canceled) AS sum_canceled
   FROM status
  GROUP BY month
)
--Calculate the churn from status_aggregate temporary table
--by dividing SUM of canceled by SUM of active, regardless of segment
SELECT month,
       1.0 * sum_canceled / sum_active AS churn_rate
  FROM status_aggregate;
```

month	churn_rate
2017-01-01	0.161687170474517
2017-02-01	0.189795918367347
2017-03-01	0.274258219727346

2.0 What is the overall churn trend since the company started? – CONT'D



3.0 Compare the churn rates between user segments

Compare the churn rates between user segments

- When looking at churn rates per user segment, we can see great disparity between segment 87 and 30.

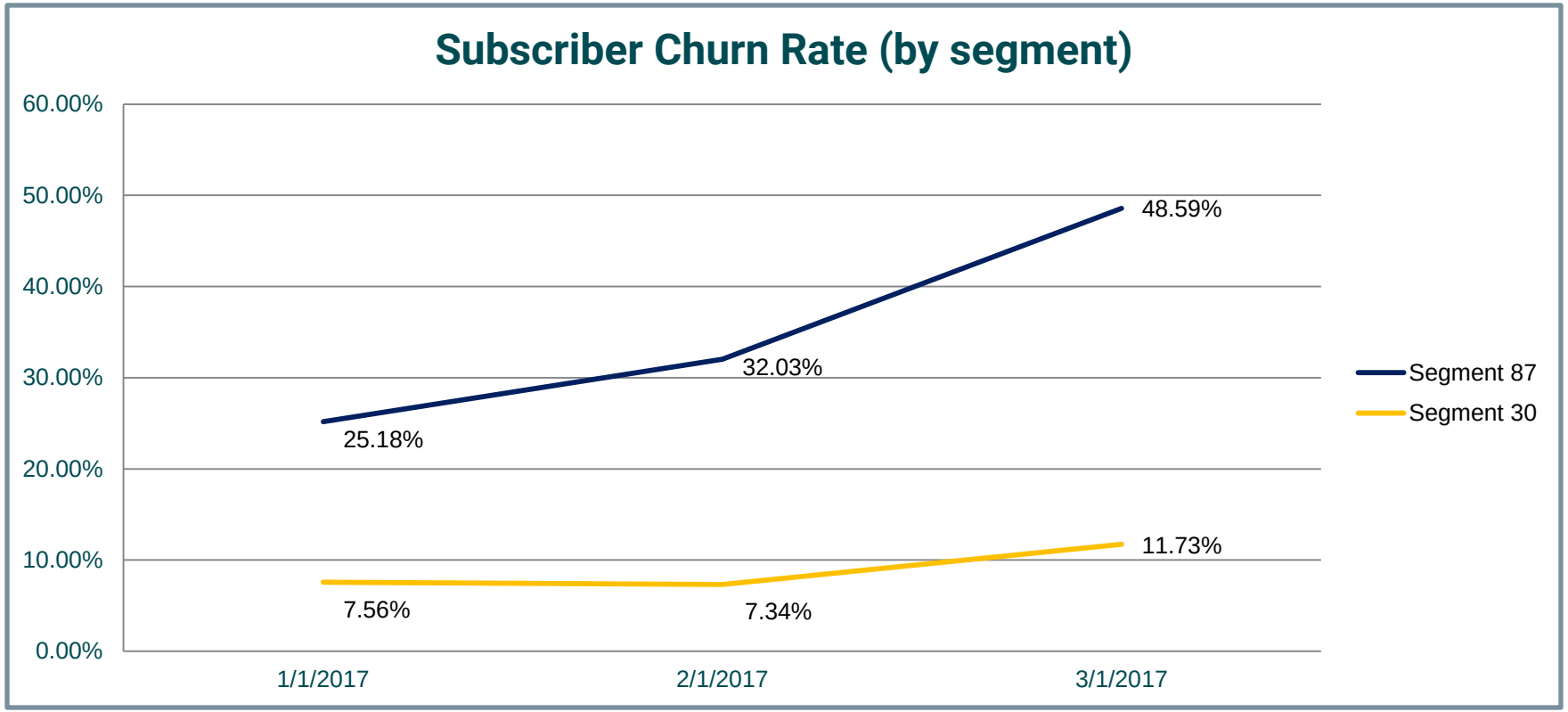
- Segment 30 is performing pretty well with 7% churn for January and February, and jumping slightly to 11% in March.

- Segment 87 started out with a 25% churn (3x segment 30's churn rate) in January. Then the churn rate increases to 32% in February and a whopping 48% in March! This is not a sustainable segment.

```sql
--3.0 Create temporary table using WITH and UNION for Jan, Feb,
--and Mar 2017
WITH months AS
(SELECT '2017-01-01' AS first_day,
        '2017-01-31' AS last_day
 UNION
 SELECT
        '2017-02-01' AS first_day,
        '2017-02-28' AS last_day
 UNION
 SELECT
        '2017-03-01' AS first_day,
        '2017-03-31' AS last_day
),
--Create temporary table cross_join to CROSS JOIN with
--subscriptions table
cross_join AS
(SELECT subscriptions.*,
        months.*
   FROM subscriptions
        CROSS JOIN months
),
--Create temporary status table to add 1's and 0's
--in order to SUM them in status_aggregate temporary table
status AS
(SELECT id,
        first_day AS month,
        CASE
          WHEN (subscription_start < first_day) AND
          (
          subscription_end > first_day OR
          subscription_end IS NULL
          ) AND
          (segment = 87) THEN 1
         ELSE 0
       END AS is_active_87,
       CASE
         WHEN (subscription_start < first_day) AND
          (
          subscription_end > first_day OR
          subscription_end IS NULL
          ) AND
          (segment = 30) THEN 1
         ELSE 0
       END AS is_active_30,
--Add is_canceled CASE statements to status temporary table
--and closing temporary status table with FROM cross_join
       CASE
         WHEN (subscription_end BETWEEN first_day AND last_day) AND
           (segment = 87) THEN 1
         ELSE 0
       END AS is_canceled_87,
       CASE
         WHEN (subscription_end BETWEEN first_day AND last_day) AND
           (segment = 30) THEN 1
         ELSE 0
       END AS is_canceled_30
  FROM cross_join
),
--Ccreate temporary status_aggregate table to SUM
--active and canceled subscriptions as columns and GROUP BY month
status_aggregate AS
(SELECT month,
        SUM(is_active_87) AS sum_active_87,
        SUM(is_active_30) AS sum_active_30,
        SUM(is_canceled_87) AS sum_canceled_87,
        SUM(is_canceled_30) AS sum_canceled_30
   FROM status
  GROUP BY month
)
--Ccalculate the churn from status_aggregate temporary table
--by dividing SUM of canceled by SUM of active, per segment
SELECT month,
       1.0 * sum_canceled_87 / sum_active_87 AS churn_rate_87,
       1.0 * sum_canceled_30 / sum_active_30 AS churn_rate_30
  FROM status_aggregate;
```

month	churn_rate_87	churn_rate_30
2017-01-01	0.251798561151079	0.0756013745704467
2017-02-01	0.32034632034632	0.0733590733590734
2017-03-01	0.485875706214689	0.11731843575419

3.0 Compare the churn rates between user segments – CONT'D



3.1 Which segment of users should the company focus on expanding?

Which segment of users should the company focus on expanding?

- Since acquiring customers is costlier than retaining subscribers, Codeflix should definitely expand on Segment 30 due to it's lower churn rate.

-However, when looking at subscriber acquisition counts by month since launching, Segment 87 is pulling just as much (or sometimes more) new subscribers as Segment 30.

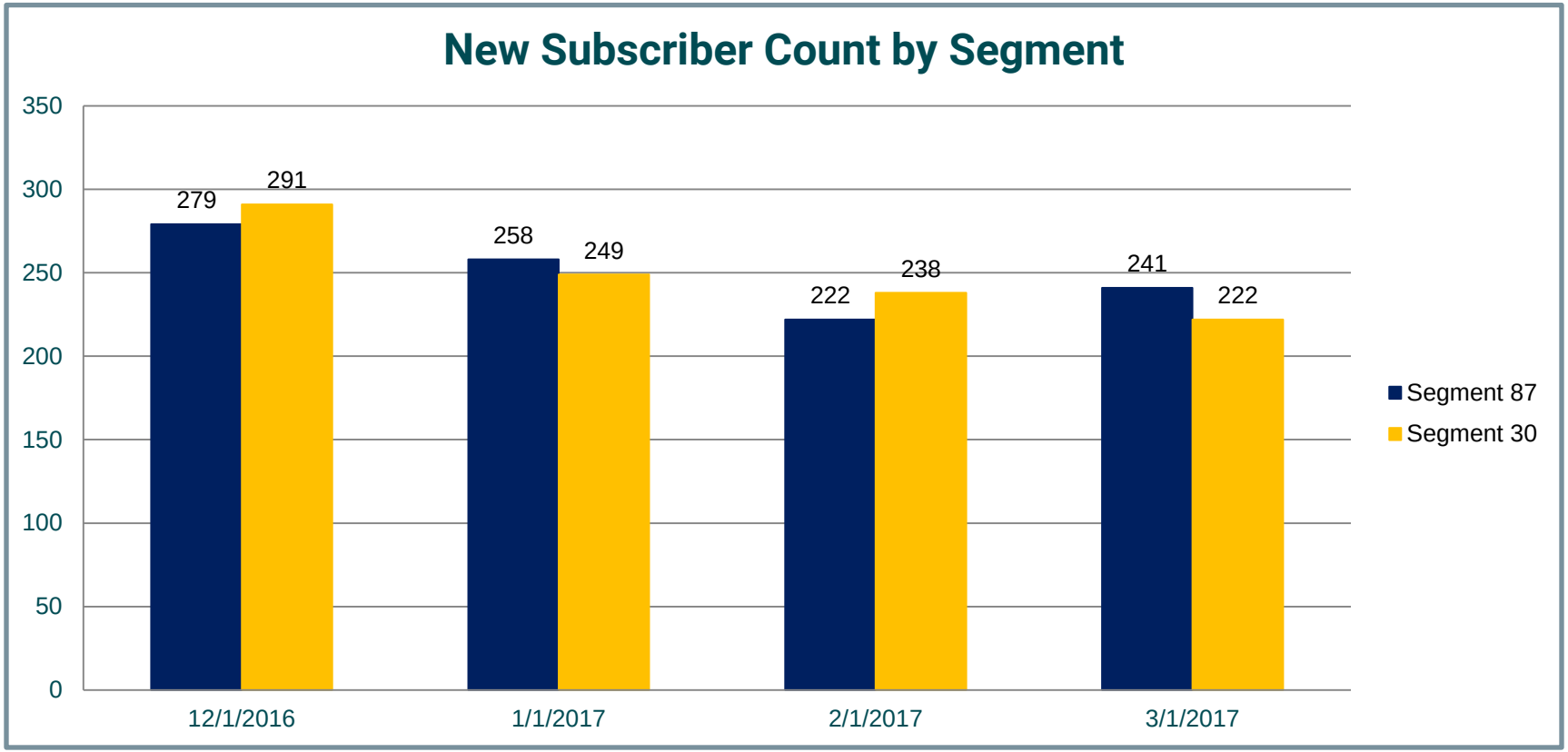
- If Codeflix utilized the same retainment strategies for both segments, then the company should dedicate more resources for segment 30.

- If the company treated subscribers differently after the initial acquisition, then Codeflix needs to apply Segment 30's retainment strategies to Segment 87 since it's just as good as a source of acquisition.

```sql
--3.1 Create temporary table using WITH and UNION for Jan, Feb,
--and Mar 2017
WITH months AS
(SELECT '2016-12-01' AS first_day,
        '2016-12-31' AS last_day
 UNION
 SELECT
        '2017-01-01' AS first_day,
        '2017-01-31' AS last_day
 UNION
 SELECT
        '2017-02-01' AS first_day,
        '2017-02-28' AS last_day
 UNION
 SELECT
        '2017-03-01' AS first_day,
        '2017-03-31' AS last_day
),
--Create temporary table cross_join to CROSS JOIN with
--subscriptions table
cross_join AS
(SELECT subscriptions.*,
        months.*
 FROM   subscriptions
        CROSS JOIN months
),
--Create temporary status table to add 1's and 0's
--in order to SUM and group new subscribers by acquisition channel
status AS
(SELECT id,
        first_day AS month,
        CASE
          WHEN (subscription_start BETWEEN first_day AND last_day)
          AND (segment = 87) THEN 1
          ELSE 0
        END AS new_subscriber_87,
        CASE
          WHEN (subscription_start BETWEEN first_day AND last_day)
          AND (segment = 30) THEN 1
          ELSE 0
        END AS new_subscriber_30
   FROM cross_join
),
--Create temporary status_aggregate table to SUM
--segment 87 and 30 subscriptions as columns and GROUP BY month
status_aggregate AS
(SELECT month,
        SUM(new_subscriber_87) AS sum_new_subscriber_87,
        SUM(new_subscriber_30) AS sum_new_subscriber_30
   FROM status
  GROUP BY month
)SELECT * FROM status_aggregate;
```

month	sum_new_subscriber_87	sum_new_subscriber_30
2016-12-01	279	291
2017-01-01	258	249
2017-02-01	222	238
2017-03-01	241	222

3.1 Which segment of users should the company focus on expanding? – CONT'D



codecademy

THANK YOU!!