# codecademy

# User Churn Analysis

Analyze Data with SQL
CafelatteHere
2023-04-21

# Analysis main points

1. Range of data given for the analysis

    ○ The company been operating since 2016-12-01;

    ○ We have enough information to calculate a churn rate for Jan, Feb, Mar of 2017;

    ○ There are two segments of users: 30 and 87;

2. Analysis of the overall churn trend since the company started;

    ○ Comparing the churn rates between user segments.

3. Further analysis of the churn rates between user segments.

    ○ Comparing the churn rates between user segments by months;
    ○ Recommendations on which segment of users should the company focus on expanding.

# 1. Range of data we can analyze

# 1.1 Range of data we can analyze

The table `subscriptions` continues data of subscriptions and unsubscriptions for Codeflix service.

- The table contains following data:
  - **id** - the subscription id
  - **subscription_start** - the start date of the subscription
  - **subscription_end** - the end date of the subscription
  - **segment** - identifies which segment the subscription owner belongs to
- SQL query [1] helped me to understand the data range we can use for analysis.
- Codeflix has been operating since 2016-12-01. We have data up until 2017-03-31 (including).
- We have enough data to calculate a churn rate for three months of 2017: January, February, March.
- We have data for two segments of users:
  - segment 30
  - segment 87

(used query [2] to check it)

| MIN(subscription_start) | MAX(subscription_start) | MIN(subscription_end) | MAX(subscription_end) |
|---|---|---|---|
| 2016-12-01 | 2017-03-30 | 2017-01-01 | 2017-03-31 |

```
–[1] Checking data range

SELECT MIN(subscription_start),
    MAX(subscription_start),
    MIN(subscription_end),
    MAX(subscription_end)
FROM subscriptions;


–[2] Segments of users

  SELECT DISTINCT segment
  FROM subscriptions;
```

# 2. Overall churn trend

# 2.1 Overall churn trend since the company started

Query [3] shows the overall churn trend since the company started for the segments 30 and 87 for January - March of 2017.
Last row of the table below shows total values for both segments over 3 months.

- Churn rate of segment 30 is more than 3 times overall lower;
- Total number of subscribers for segment 30 is almost 20% bigger than for segment 87.

```
-- Query [3]
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
),
cross_join AS (SELECT *
FROM months
CROSS JOIN subscriptions),

status AS (SELECT
  id,
  segment,
  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,
  CASE WHEN subscription_end BETWEEN first_day AND last_day
    THEN 1
    ELSE 0
  END as is_canceled
FROM cross_join)

SELECT month, segment,
  SUM(is_active),
  SUM(is_canceled),
  (SUM(1.0 * is_canceled) / SUM(is_active)) as churn_rate_by_segment
FROM status
GROUP BY  segment;
```

| period | segment | active | canceled | churn_rate_by_segment |
|---|---|---|---|---|
| 2017-01-01 - 2017-03-31 | 30 | 1525 | 144 | 0.0944 |
| 2017-01-01 - 2017-03-31 | 87 | 1271 | 476 | 0.374 |
| 2017-01-01 - 2017-03-31 | n/a | 2796 | 620 | 0.221 |

*Churn_rate_by_segment results are rounded up to hundredth place.*

# 3. Which segment of users should the company focus on expanding?

# 3.1 Comparison of the churn rates between user segments

Comparison of the same data by each month (Jan, Feb, Mar, 2017)

Query used to get the results: [4] (see Appendix 1)

- Overall, segment 30 is showing better churn rate, we should focus on this segment.
- Or investigate why there is so much cancellations in segment 87 if we want to keep it
- Both segments showed a spike of cancellations for Mar 2017
- We might need to investigate what caused it

| month | segment | active | canceled | churn_rate_by_segment |
|---|---|---|---|---|
| 2017-01-01 | 30 | 291 | 22 | 0.076 |
| 2017-01-01 | 87 | 278 | 70 | 0.252 |
| 2017-02-01 | 30 | 518 | 38 | 0.073 |
| 2017-02-01 | 87 | 462 | 148 | 0.320 |
| 2017-03-01 | 30 | 716 | 84 | 0.117 |
| 2017-03-01 | 87 | 531 | 258 | 0.486 |

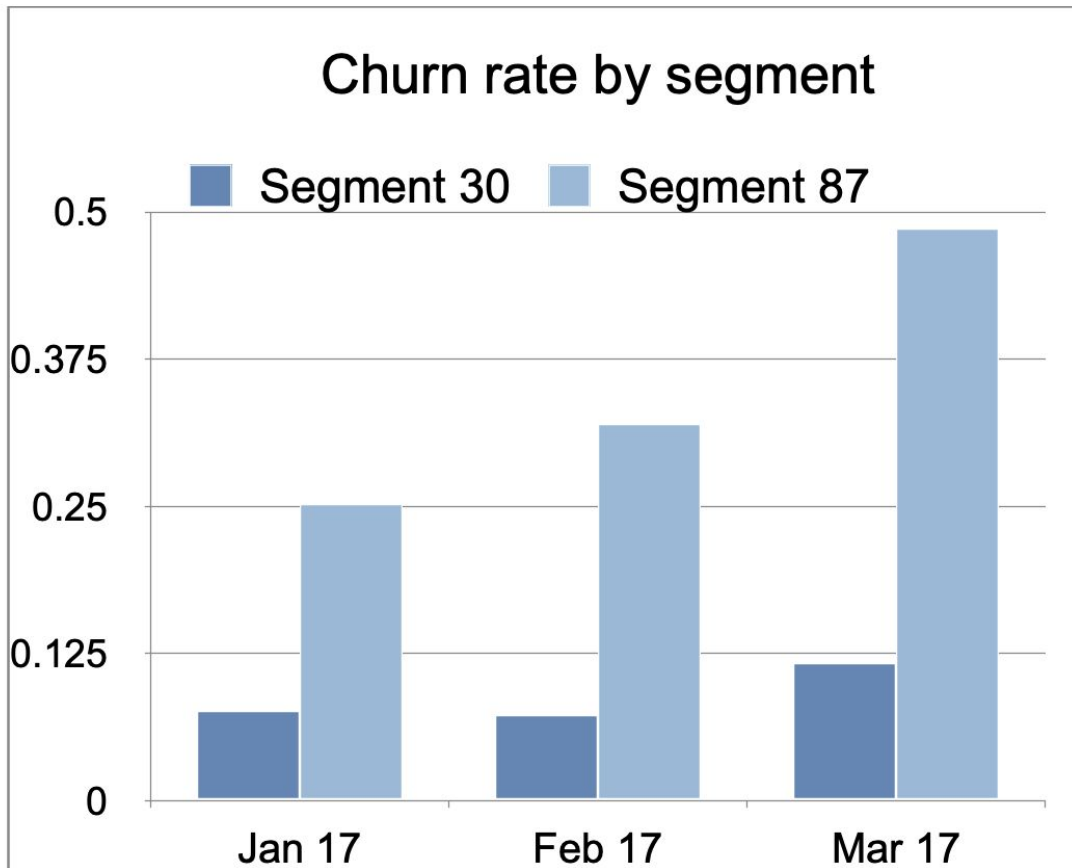*Churn_rate_by_segment results are rounded up to hundredth place.*

# 3.2 Comparison of the churn rates between user segments

Comparison of churn rates by each month (Jan, Feb, Mar, 2017)

Churn rates from the previous slide represented in a visual form (bar graph)

| month | segment | churn_rate_by_segment |
|---|---|---|
| 2017-01-01 | 30 | 0.076 |
| 2017-01-01 | 87 | 0.252 |
| 2017-02-01 | 30 | 0.073 |
| 2017-02-01 | 87 | 0.320 |
| 2017-03-01 | 30 | 0.117 |
| 2017-03-01 | 87 | 0.486 |

*Churn_rate_by_segment results are rounded up to hundredth place.*



Churn rate by segment

# Appendix 1

```sql
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
),
cross_join AS (SELECT *
FROM months
CROSS JOIN subscriptions),

status AS (SELECT
  id,
  segment,
  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,
 CASE WHEN subscription_end BETWEEN first_day AND last_day
    THEN 1
    ELSE 0
  END as is_canceled
FROM cross_join)

SELECT month, segment,
  SUM(is_active),
  SUM(is_canceled),
  (SUM(1.0 * is_canceled) / SUM(is_active)) as churn_rate_by_segment
FROM status
GROUP BY  month,segment;
```

< Query [4]