CaseStudy_TelcoCustomerChurn

Jinxiu

2025-11-28

Phase 1: Business Questions

- What is the current overall churn rate? In which customer groups is it higher?
- Which factors are strongly related to churn?
- If we need a retention outreach list (limited budget), who should be prioritized and why?

Phase 2: Data Structure

2.1 Dataset Source & Scope

The dataset **Telco Customer Churn** is publicly available in Kaggle (https://www.kaggle.com/datasets/blastchar/telco-customer-churn/data). The dataset contains **7032 customer records** with **21 fields**, where **each row represents a single customer snapshot** at a specific time point. The target variable of this analysis is a binary field *Churn*.

2.2 Schema & Key Fields

- **Database**: churn_db (MySQL)
- **Table**: customer_churn (raw data)
- **Grain**: one row per customer
- **Primary key**: customerID (unique, no duplicates found)
- **Target variable**: Churn (Yes/No)
- **Key numeric metrics**: tenure (months), MonthlyCharges, TotalCharges

2.3 Variable Groups

Variables are grouped by business terminology to support segmentation and actionable insights:

(A) Customer profile

- gender, SeniorCitizen, Partner, Dependents

(B) Account & billing

- Contract, PaperlessBilling, PaymentMethod

(C) Services

- PhoneService, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies

(D) Tenure & charges

- tenure, MonthlyCharges, TotalCharges

2.4 Quick Data Check

```
# load required package
library(DBI)
library(RMariaDB)
library(dotenv)
library(here)
```

```
## here() starts at D:/CaseStudy_TelcoCustomerChurn/r_report
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.1     v readr     2.1.6
## v ggplot2   4.0.1     v stringr   1.6.0
## v lubridate 1.9.4     v tibble    3.3.0
## v purrr     1.2.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

```r
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
library(patchwork)
```

```r
# connect with SQL
load_dot_env(file = '.env')

con <- dbConnect(
  MariaDB(),
  host = Sys.getenv("DB_HOST"),
  port = as.integer(Sys.getenv("DB_PORT")),
  dbname = Sys.getenv("DB_NAME"),
  user = Sys.getenv("DB_USER"),
  password = Sys.getenv("DB_PASSWORD")
```

```
)
# test: check the table
dbListTables(con)
```

```
## [1] "customer_churn"
```

A. Overview

There are 7032 rows and 21 fields. Unique primary key is *customerID* with no duplication.

```
# obtain schema information
dbGetQuery(con,"
  SHOW COLUMNS FROM customer_churn;
")
```

```
##               Field   Type Null Key Default Extra
## 1        customerID   text  YES        <NA>
## 2            gender   text  YES        <NA>
## 3     SeniorCitizen    int  YES        <NA>
## 4           Partner   text  YES        <NA>
## 5        Dependents   text  YES        <NA>
## 6            tenure    int  YES        <NA>
## 7      PhoneService   text  YES        <NA>
## 8     MultipleLines   text  YES        <NA>
## 9   InternetService   text  YES        <NA>
## 10   OnlineSecurity   text  YES        <NA>
## 11     OnlineBackup   text  YES        <NA>
## 12 DeviceProtection   text  YES        <NA>
## 13      TechSupport   text  YES        <NA>
## 14      StreamingTV   text  YES        <NA>
## 15  StreamingMovies   text  YES        <NA>
## 16         Contract   text  YES        <NA>
## 17 PaperlessBilling   text  YES        <NA>
## 18    PaymentMethod   text  YES        <NA>
## 19   MonthlyCharges double  YES        <NA>
## 20     TotalCharges double  YES        <NA>
## 21            Churn   text  YES        <NA>
```

```
# check total count of rows and unique primary keys
dbGetQuery(con,"
  SELECT COUNT(*) AS num_rows,
         COUNT(DISTINCT customerID) AS num_unique_customer,
         COUNT(*) - COUNT(DISTINCT customerID) AS num_duplicate_customer
  FROM customer_churn;
")
```

```
##   num_rows num_unique_customer num_duplicate_customer
## 1     7032                7032                      0
```

B. Target variable

Target variable *Churn* is binary, only including Yes and No. There exits slightly imbalance but acceptable business ratio.

```
# check distribution of Churn
dbGetQuery(con, "
  SELECT Churn, COUNT(*) AS num,
         ROUND(
           COUNT(*) * 100 / (
             SELECT COUNT(*)
             FROM customer_churn
           ), 2
         ) AS percent
  FROM customer_churn
  GROUP BY Churn;
")
```

```
##   Churn  num percent
## 1    No 5163   73.42
## 2   Yes 1869   26.58
```

C. Numeric fields

All numeric fields have no missing values within a reasonable range.

```
# check key numeric fields' missing
dbGetQuery(con, "
  SELECT
```

```
    SUM(CASE WHEN tenure IS NULL THEN 1 ELSE 0 END) AS tenure_missing,
    SUM(CASE WHEN MonthlyCharges IS NULL THEN 1 ELSE 0 END) AS moncha_missing,
    SUM(CASE WHEN TotalCharges IS NULL THEN 1 ELSE 0 END) AS totcha_missing
  FROM customer_churn;
")
```

```
##   tenure_missing moncha_missing totcha_missing
## 1              0              0              0
```

```
# check range of numeric fields
dbGetQuery(con, "
  SELECT
    MIN(tenure) AS tenure_min, MAX(tenure) AS tenure_max,
    MIN(MonthlyCharges) AS moncha_min, MAX(MonthlyCharges) AS moncha_max,
    MIN(TotalCharges) AS totcha_min, MAX(TotalCharges) AS totcha_max
  FROM customer_churn;
")
```

```
##   tenure_min tenure_max moncha_min moncha_max totcha_min totcha_max
## 1          1         72      18.25     118.75       18.8     8684.8
```

```
# check the consistency between TotalCharges and tenure
dbGetQuery(con, "
  SELECT COUNT(*) AS suspicious_totcha
  FROM customer_churn
  WHERE TotalCharges = 0
  AND tenure > 0;
")
```

```
##   suspicious_totcha
## 1                 0
```

D. Categorical fields

Key textual fields have meaningful categories.

```
dbGetQuery(con, "
  SELECT Contract, COUNT(*) AS num
  FROM customer_churn
```

```
  GROUP BY Contract
  ORDER BY num DESC;
")
```

```
##         Contract  num
## 1 Month-to-month 3875
## 2       Two year 1685
## 3       One year 1472
```

```
dbGetQuery(con, "
  SELECT PaymentMethod, COUNT(*) AS num
  FROM customer_churn
  GROUP BY PaymentMethod
  ORDER BY num DESC;
")
```

```
##               PaymentMethod  num
## 1          Electronic check 2365
## 2              Mailed check 1604
## 3 Bank transfer (automatic) 1542
## 4   Credit card (automatic) 1521
```

```
dbGetQuery(con, "
  SELECT InternetService, COUNT(*) AS num
  FROM customer_churn
  GROUP BY InternetService
  ORDER BY num DESC;
")
```

```
##   InternetService  num
## 1     Fiber optic 3096
## 2             DSL 2416
## 3              No 1520
```

Phase 3: Data Cleaning

3.1 Data Type & Readability

- Reassign label of *SeniorCitizen* into more understandable yes(=1) and no(=0)
- Convert all categorical fields into factor type

3.2 Missing/Abnormal Values & Empty Strings

- No NA or NULL values in data
- Key numeric fields in the reasonable range

3.3 Repetition & Uniqueness

- No repetition
- Checking consistency of *TotalCharges* and being up to standard

3.4 Categorical Field Standardization

- Checking levels of key categorical fields
- No empty strings, excessive spaces, and inconsistent letter case
- Retain business semantics such as 'No internet service' and 'No phone service'

3.5 Code Implementation

```
# store raw data into dataframe

df_churn <- dbGetQuery(con, "
              SELECT *
              FROM customer_churn;
            ")
```

```
# check raw data
glimpse(df_churn)
```

```
## Rows: 7,032
## Columns: 21
## $ customerID       <chr> "7590-VHVEG", "5575-GNVDE", "3668-QPYBK", "7795-CFOCW~
## $ gender           <chr> "Female", "Male", "Male", "Male", "Female", "Female",~
## $ SeniorCitizen    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ Partner          <chr> "Yes", "No", "No", "No", "No", "No", "No", "No", "Yes~
## $ Dependents       <chr> "No", "No", "No", "No", "No", "No", "Yes", "No", "No"~
## $ tenure           <int> 1, 34, 2, 45, 2, 8, 22, 10, 28, 62, 13, 16, 58, 49, 2~
## $ PhoneService     <chr> "No", "Yes", "Yes", "No", "Yes", "Yes", "Yes", "No", ~
## $ MultipleLines    <chr> "No phone service", "No", "No", "No phone service", "~
## $ InternetService  <chr> "DSL", "DSL", "DSL", "DSL", "Fiber optic", "Fiber opt~
```

```
## $ OnlineSecurity   <chr> "No", "Yes", "Yes", "Yes", "No", "No", "No", "Yes", "~
## $ OnlineBackup     <chr> "Yes", "No", "Yes", "No", "No", "No", "Yes", "No", "N~
## $ DeviceProtection <chr> "No", "Yes", "No", "Yes", "No", "Yes", "No", "No", "Y~
## $ TechSupport      <chr> "No", "No", "No", "Yes", "No", "No", "No", "No", "Yes~
## $ StreamingTV      <chr> "No", "No", "No", "No", "No", "Yes", "Yes", "No", "Ye~
## $ StreamingMovies  <chr> "No", "No", "No", "No", "No", "Yes", "No", "No", "Yes~
## $ Contract         <chr> "Month-to-month", "One year", "Month-to-month", "One ~
## $ PaperlessBilling <chr> "Yes", "No", "Yes", "No", "Yes", "Yes", "Yes", "No", ~
## $ PaymentMethod    <chr> "Electronic check", "Mailed check", "Mailed check", "~
## $ MonthlyCharges   <dbl> 29.85, 56.95, 53.85, 42.30, 70.70, 99.65, 89.10, 29.7~
## $ TotalCharges     <dbl> 29.85, 1889.50, 108.15, 1840.75, 151.65, 820.50, 1949~
## $ Churn            <chr> "No", "No", "Yes", "No", "Yes", "Yes", "No", "No", "Y~
```

```
# check NULL value
is.null(df_churn)
```

```
## [1] FALSE
```

```
# check missing value
unique(is.na(df_churn))
```

```
##      customerID gender SeniorCitizen Partner Dependents tenure PhoneService
## [1,]      FALSE  FALSE         FALSE   FALSE      FALSE  FALSE        FALSE
##      MultipleLines InternetService OnlineSecurity OnlineBackup DeviceProtection
## [1,]         FALSE           FALSE          FALSE        FALSE            FALSE
##      TechSupport StreamingTV StreamingMovies Contract PaperlessBilling
## [1,]       FALSE       FALSE           FALSE    FALSE            FALSE
##      PaymentMethod MonthlyCharges TotalCharges Churn
## [1,]         FALSE          FALSE        FALSE FALSE
```

```
df_churn_clean <- df_churn %>%
  # reassign SeniorCitizen into 'Yes' and 'No'
  mutate(SeniorCitizen =
           ifelse(SeniorCitizen == 1, 'Yes', 'No')
         ) %>%
  # convert character fields into factor
  mutate(across(c('gender', 'SeniorCitizen', 'Partner',
                  'Dependents', 'PhoneService', 'MultipleLines',
                  'InternetService', 'OnlineSecurity', 'OnlineBackup',
                  'DeviceProtection', 'TechSupport', 'StreamingTV',
```

```
                  'StreamingMovies', 'Contract', 'PaperlessBilling',
                  'PaymentMethod', 'Churn')
                , factor)
         )
```

```
# check cleaned data
summary(df_churn_clean)
```

```
##   customerID           gender     SeniorCitizen Partner    Dependents
##  Length:7032        Female:3483   No :5890      No :3639   No :4933
##  Class :character   Male  :3549   Yes:1142      Yes:3393   Yes:2099
##  Mode  :character
##
##
##
##      tenure      PhoneService          MultipleLines     InternetService
##  Min.   : 1.00   No : 680     No              :3385   DSL        :2416
##  1st Qu.: 9.00   Yes:6352     No phone service: 680   Fiber optic:3096
##  Median :29.00                Yes             :2967   No         :1520
##  Mean   :32.42
##  3rd Qu.:55.00
##  Max.   :72.00
##              OnlineSecurity              OnlineBackup
##  No                 :3497   No                 :3087
##  No internet service:1520   No internet service:1520
##  Yes                :2015   Yes                :2425
##
##
##
##             DeviceProtection              TechSupport
##  No                 :3094    No                 :3472
##  No internet service:1520    No internet service:1520
##  Yes                :2418    Yes                :2040
##
##
##
##               StreamingTV              StreamingMovies           Contract
##  No                 :2809   No                 :2781   Month-to-month:3875
##  No internet service:1520   No internet service:1520   One year      :1472
##  Yes                :2703   Yes                :2731   Two year      :1685
```

```
##
##
##
##  PaperlessBilling                   PaymentMethod  MonthlyCharges
##  No :2864         Bank transfer (automatic):1542   Min.   : 18.25
##  Yes:4168         Credit card (automatic)  :1521   1st Qu.: 35.59
##                   Electronic check         :2365   Median : 70.35
##                   Mailed check             :1604   Mean   : 64.80
##                                                    3rd Qu.: 89.86
##                                                    Max.   :118.75
##   TotalCharges    Churn
##  Min.   :  18.8   No :5163
##  1st Qu.: 401.4   Yes:1869
##  Median :1397.5
##  Mean   :2283.3
##  3rd Qu.:3794.7
##  Max.   :8684.8
```

After cleaning and type standardization, the analysis-ready dataset is stored as *df_churn_clean* and used for subsequent feature engineering and EDA.

Phase 4: Feature Engineering

Feature engineering is used for enhancing interpretable cluseter analysis and follow-up action recommendations.

4.1 Definition & Significance

- **tenure_bucket**: bins *tenure* into lifecycle stages to compare churn across customer maturity and support targeted retention timing
- **charges_bucket**: quartile-based monthly charge segments to identify high-value customers and price-sensitive groups
- **auto_pay**: flags automatic payment methods for churn-risk comparison
- **has_internet**: identifies whether customers subscribe to an internet service to contextualize 'No internet service' fields
- **services_count**: counts subscribed add-on services as a proxy for service bundling/switching cost

4.2 Code Implementation

```
# binning tenure
df_churn_fe <- df_churn_clean %>%
  mutate(
    tenure_bucket = case_when(
      tenure <= 6 ~ '0-6 months',
      tenure <= 12 ~ '7-12 months',
      tenure <= 24 ~ '13-24 months',
      tenure <= 48 ~ '25-48 months',
      TRUE ~ '49-72 months'
    ),
    tenure_bucket = factor(
      tenure_bucket,
      levels = c('0-6 months', '7-12 months',
                 '13-24 months', '25-48 months', '49-72 months')
    )
  )
```

```
# quartile-based binning monthly charges
q <- quantile(df_churn_fe$MonthlyCharges,
              probs = c(0, 0.25, 0.5, 0.75, 1),
              na.rm = TRUE)

df_churn_fe <- df_churn_fe %>%
  mutate(
    charges_bucket = cut(
      MonthlyCharges,
      breaks = unique(q),
      labels = c('Low', 'Mid-Low', 'Mid-High', 'High'),
      include.lowest = TRUE
    )
  )
```

```
# label whether the charge is automatically deducted
df_churn_fe <- df_churn_fe %>%
  mutate(
    auto_pay = if_else(str_detect(as.character(PaymentMethod), 'automatic'),
                       'Automatic', 'Non-automatic'),
    auto_pay = factor(auto_pay,
```

```
                        levels = c('Non-automatic', 'Automatic')
                        )
  )
```

```
# label whether there is internet service
df_churn_fe <- df_churn_fe %>%
  mutate(
    has_internet = ifelse(as.character(InternetService) == 'No',
                          'No provider-support', 'Provider-support'),
    has_internet = factor(has_internet,
                          levels = c('No provider-support', 'Provider-support')
                          )
  )
```

```
# summarize count of purchasing services per customer
service_cols <- c(
  'PhoneService', 'MultipleLines',
  'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
  'TechSupport', 'StreamingTV', 'StreamingMovies'
)

df_churn_fe <- df_churn_fe %>%
  mutate(
    services_count = rowSums(
      across(
        all_of(service_cols), ~as.integer(as.character(.x) == 'Yes')
        )
      )
  )
```

```
# sanity check summary
check_table <- list(
  tenure_bucket = table(df_churn_fe$tenure_bucket),
  charges_bucket = table(df_churn_fe$charges_bucket),
  auto_pay = table(df_churn_fe$auto_pay),
  has_internet = table(df_churn_fe$has_internet),
  services_count = summary(df_churn_fe$services_count)
)
check_table
```

```
## $tenure_bucket
##
##   0-6 months  7-12 months 13-24 months 25-48 months 49-72 months
##         1470          705         1024         1594         2239
##
## $charges_bucket
##
##      Low  Mid-Low Mid-High     High
##     1758     1761     1755     1758
##
## $auto_pay
##
## Non-automatic     Automatic
##          3969          3063
##
## $has_internet
##
## No provider-support    Provider-support
##                1520                5512
##
## $services_count
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   3.000   3.363   5.000   8.000
```

Phase 5: EDA & Segmentation

This stage is to identify the key differentiated populations for churn rate, explore variables related to churn, as well as generate actionable segmented groups.
With **Churn = Yes** as the focus, the main approach is to compare the churn rate between different populations.

5.1 Ovarall Churn Snapshot

The dataset contains 7032 customers, where count of churned customers (*Churn* = Yes)is 1869 (26.6%) and count of non-churned customers is 5163 (73.4%). The overall displays certain class imbalance. But it does not affects subsequent comparison analysis of using churn rate for different customer groups, which is used to identify high-risk segment groups.

```r
churn_table <- df_churn_fe %>%
  group_by(Churn) %>%
  summarise(cnt_customer = n()) %>%
  mutate(
    percent = round(
      cnt_customer / nrow(df_churn_fe)
    , 4)
  )
churn_table
```

```
## # A tibble: 2 x 3
##   Churn cnt_customer percent
##   <fct>        <int>   <dbl>
## 1 No            5163   0.734
## 2 Yes           1869   0.266
```

5.2 Churn by Lifecycle & Price

A. Tenure lifecycle segmentation

```r
# distribution of tenure segment
tenure_churn_table <- df_churn_fe %>%
  group_by(tenure_bucket) %>%
  summarise(tenure_cnt = n()) %>%
  mutate(
    tenure_pct = round(
      tenure_cnt / nrow(df_churn_fe)
    , 4)
  )
tenure_churn_table
```

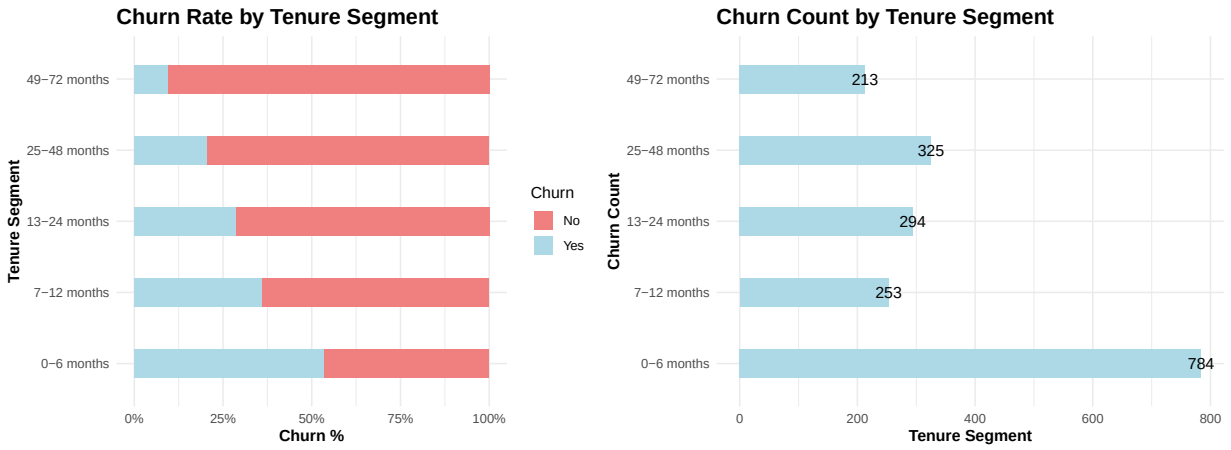
```
## # A tibble: 5 x 3
##   tenure_bucket tenure_cnt tenure_pct
##   <fct>              <int>      <dbl>
## 1 0-6 months          1470      0.209
## 2 7-12 months          705      0.100
## 3 13-24 months        1024      0.146
## 4 25-48 months        1594      0.227
## 5 49-72 months        2239      0.318
```

```r
# churn rate under each tenure segment
p1 <- df_churn_fe %>%
  count(tenure_bucket, Churn) %>%
  group_by(tenure_bucket) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(
    aes(x = percent,
        y = tenure_bucket,
        fill = Churn)
    ) +
  geom_col(position = 'fill',
           width = 0.4,
           linewidth = 1.5
    ) +
  scale_x_continuous(labels = percent) +
  scale_fill_manual(
    values = c(
      'No' = 'lightcoral',
      'Yes' = 'lightblue'
      )
    ) +
  labs(
    title = 'Churn Rate by Tenure Segment',
    x = 'Churn %',
    y = 'Tenure Segment',
    fill = 'Churn'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
```

```r
# churn count under each tenure segment
p2 <- df_churn_fe %>%
  filter(Churn == 'Yes') %>%
  count(tenure_bucket) %>%
  ggplot(
    aes(x = n,
```

```
        y = tenure_bucket)
    ) +
  geom_col(fill = 'lightblue',
           width = 0.4,
           linewidth = 1.5
    ) +
  geom_text(aes(label = n)) +
  labs(
    title = 'Churn Count by Tenure Segment',
    x = 'Tenure Segment',
    y = 'Churn Count'
  ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
```

```
(p1 | p2) +
  plot_layout(widths = c(1, 1.3))
```



In term of customer lifecycle, churn concentrates mainly on the early stage: the churn rate in "0-6 months" is the highest (50%), and then displays decreasing trend with tenure increasing. This means that retention will enhance over time.

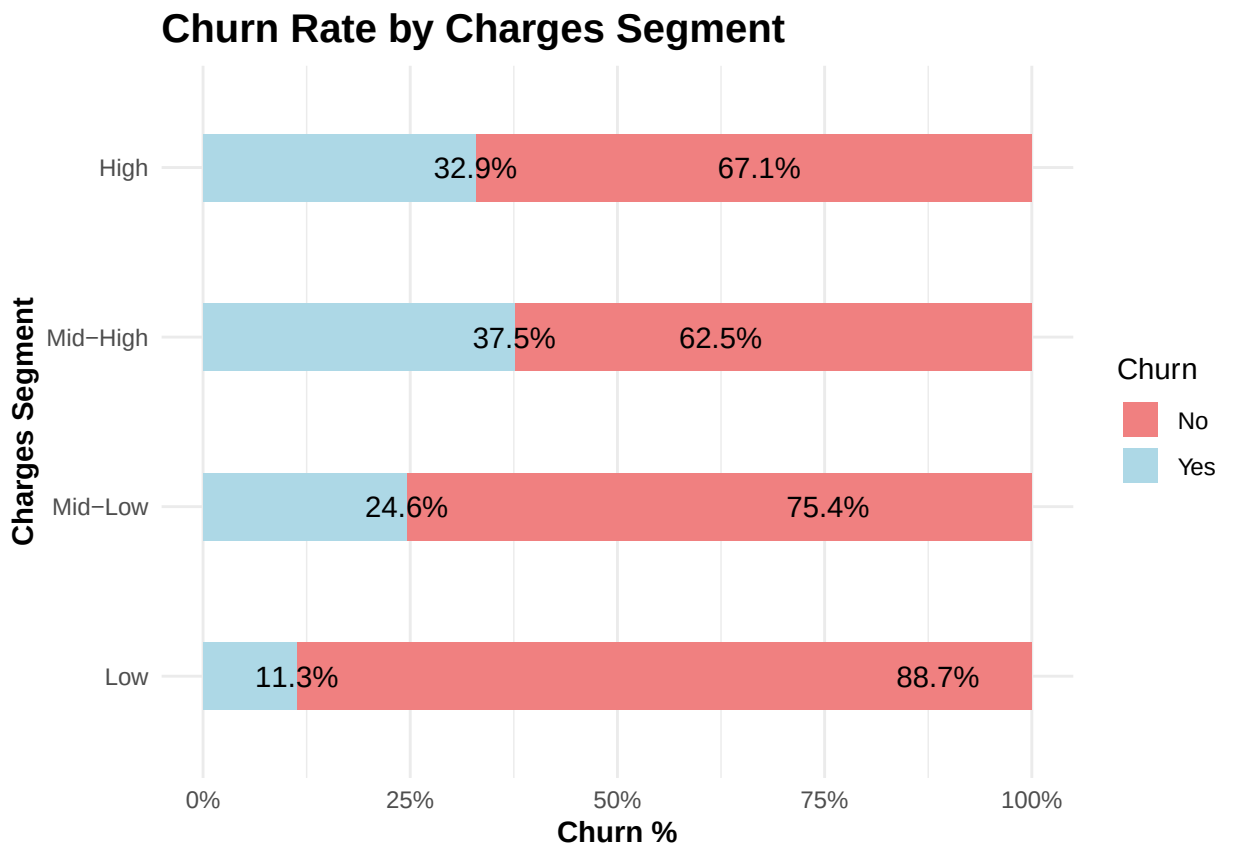
Considering that different tenure segment has different customer base, the right picture further figures out churn counts under each group: churn count in "0-6 months" is the maximum, which is the primary source of the overall churn. Moreover, although churn rate in "25-48 months" is low, its customer base is relatively large that also contributes a considerable number of churn.

Therefore, if the goal is "maximize to reduce churned customer", it is necessary to prioritize to cover **early tenure segments**. If the goal is "improve old customer retention", it can further intervene in medium-term customer with the differentiation strategy.

B. Pricing segmentation

```
# churn rate under each charges bucket
p3 <- df_churn_fe %>%
  count(charges_bucket, Churn) %>%
  group_by(charges_bucket) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(
    aes(x = percent,
        y = charges_bucket,
        fill = Churn)
    ) +
  geom_col(position = 'fill',
           width = 0.4, linewidth = 1.5) +
  geom_text(
    aes(label = paste0(round(percent*100, 1), '%'))
    , vjust = 0.5) +
  scale_x_continuous(labels = percent) +
  scale_fill_manual(
    values = c(
      'No' = 'lightcoral',
      'Yes' = 'lightblue'
      )
    ) +
  labs(
    title = 'Churn Rate by Charges Segment',
    x = 'Churn %',
    y = 'Charges Segment',
    fill = 'Churn'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
```

```
p3
```



By monthly charge grouping (charges_bucket, quartile grouping), it is observed that churn rate increases as the monthly charge level rises: the churn rate is the lowest in the "Low" group (about 11%), while churn rate is significantly higher in the "Mid-High"/"High" group. Among them, the churn rate of "Mid-High" is slightly higher than that of "High", indicating that high charges is not the only explanation. It may be related to differences in package structure, service type (such as *InternetService*), or payment method.

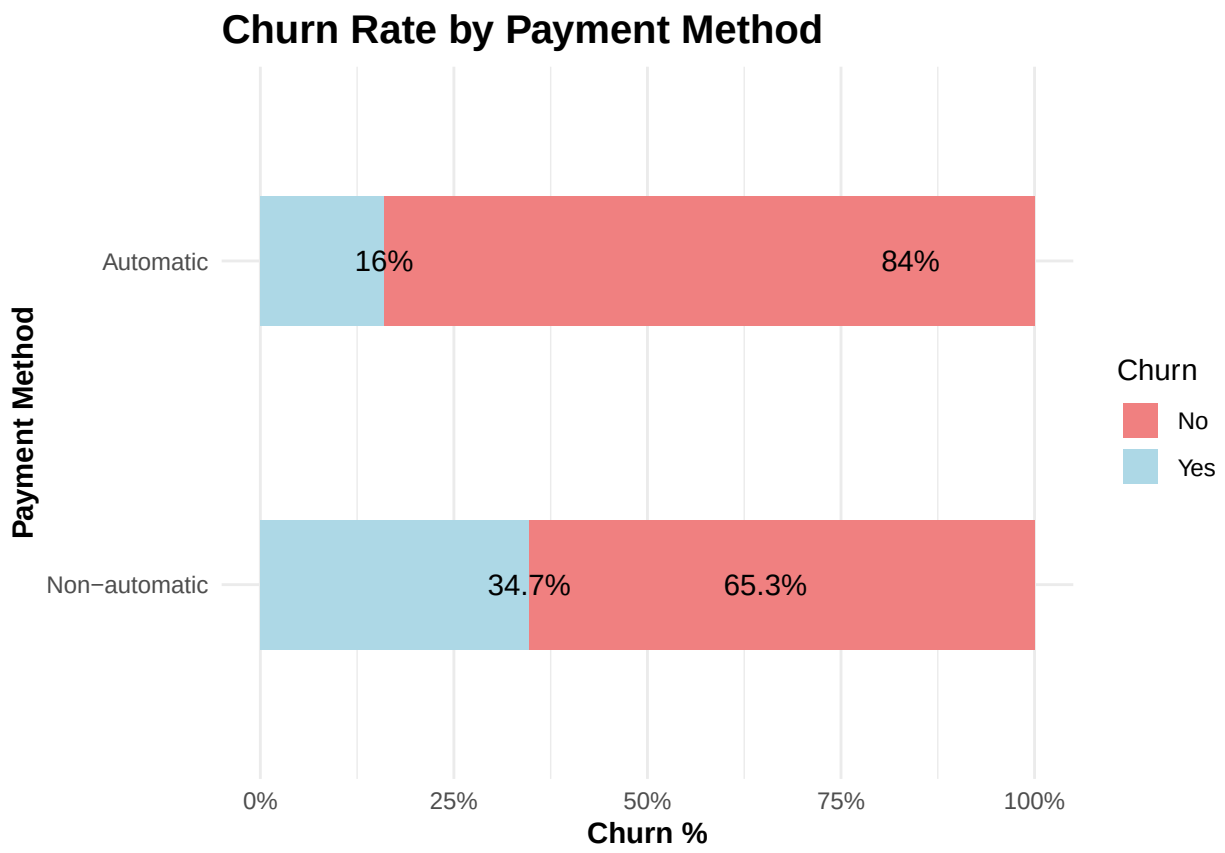
Therefore, through cross-grouping (such as *charges_bucket* x *auto_pay*) the subsequent analysis verifies whether this phenomenon id driven by different customer structure.

5.3 Payment Choice & Service Context

A. Automatic payment

```
# churn rate between automatic and other payment
p4 <- df_churn_fe %>%
  count(auto_pay, Churn) %>%
```

```
  group_by(auto_pay) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(
    aes(x = percent,
        y = auto_pay,
        fill = Churn)
    ) +
  geom_col(position = 'fill',
           width = 0.4, linewidth = 1.5) +
  geom_text(
    aes(label = paste0(round(percent*100, 1), '%'))
    , vjust = 0.5) +
  scale_x_continuous(labels = percent) +
  scale_fill_manual(
    values = c(
      'No' = 'lightcoral',
      'Yes' = 'lightblue'
      )
    ) +
  labs(
    title = 'Churn Rate by Payment Method',
    x = 'Churn %',
    y = 'Payment Method',
    fill = 'Churn'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
p4
```



The churn rate of customers with automatic fee deduction is significantly lower than those without automatic fee deduction (about 16% vs 34.7%). The two groups of customers are of similar size, so this difference has strong business significance: the renewal friction related to payment methods may be an important factor affecting retention.

In terms of business, an executable strategy can adopt "guiding non-automatic deduction customers to migrate to automatic deduction", and priority should be given to combining with tenure (new customers) or high-monthly charge groups for targeted reach to enhance conversion revenue.

B. Internet service subscription

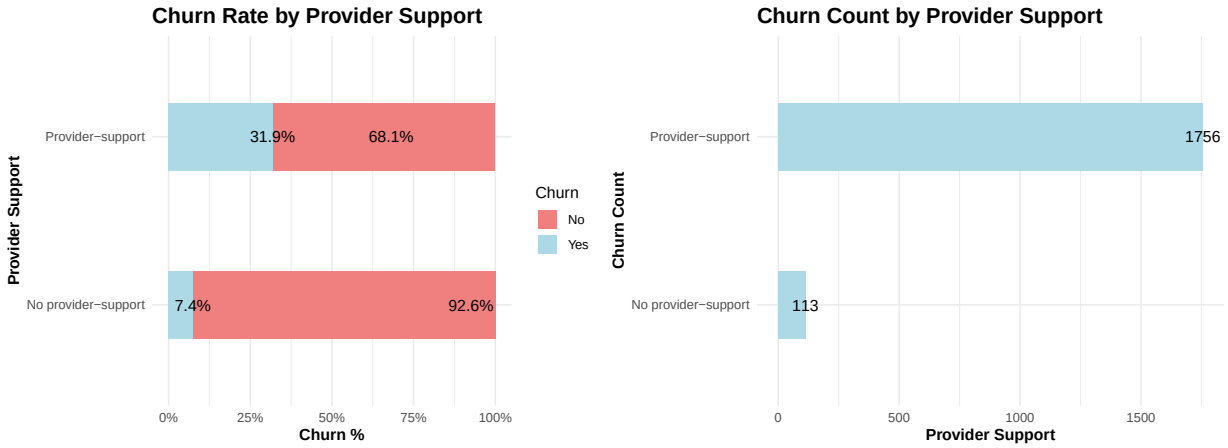
```
# churn rate between provider-support and no provider-support
p5 <- df_churn_fe %>%
  count(has_internet, Churn) %>%
  group_by(has_internet) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(
    aes(x = percent,
        y = has_internet,
```

```
        fill = Churn)
    ) +
  geom_col(position = 'fill',
           width = 0.4, linewidth = 1.5) +
  geom_text(
    aes(label = paste0(round(percent*100, 1), '%'))
    , vjust = 0.5) +
  scale_x_continuous(labels = percent) +
  scale_fill_manual(
    values = c(
      'No' = 'lightcoral',
      'Yes' = 'lightblue'
      )
    ) +
  labs(
    title = 'Churn Rate by Provider Support',
    x = 'Churn %',
    y = 'Provider Support',
    fill = 'Churn'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
```

```
# churn count between provider-support and no provider-support
p6 <- df_churn_fe %>%
  filter(Churn == 'Yes') %>%
  count(has_internet) %>%
  ggplot(
    aes(x = n,
        y = has_internet)
    ) +
  geom_col(fill = 'lightblue',
           width = 0.4,
           linewidth = 1.5
    ) +
  geom_text(aes(label = n)) +
```

```
  labs(
    title = 'Churn Count by Provider Support',
    x = 'Provider Support',
    y = 'Churn Count'
  ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
```

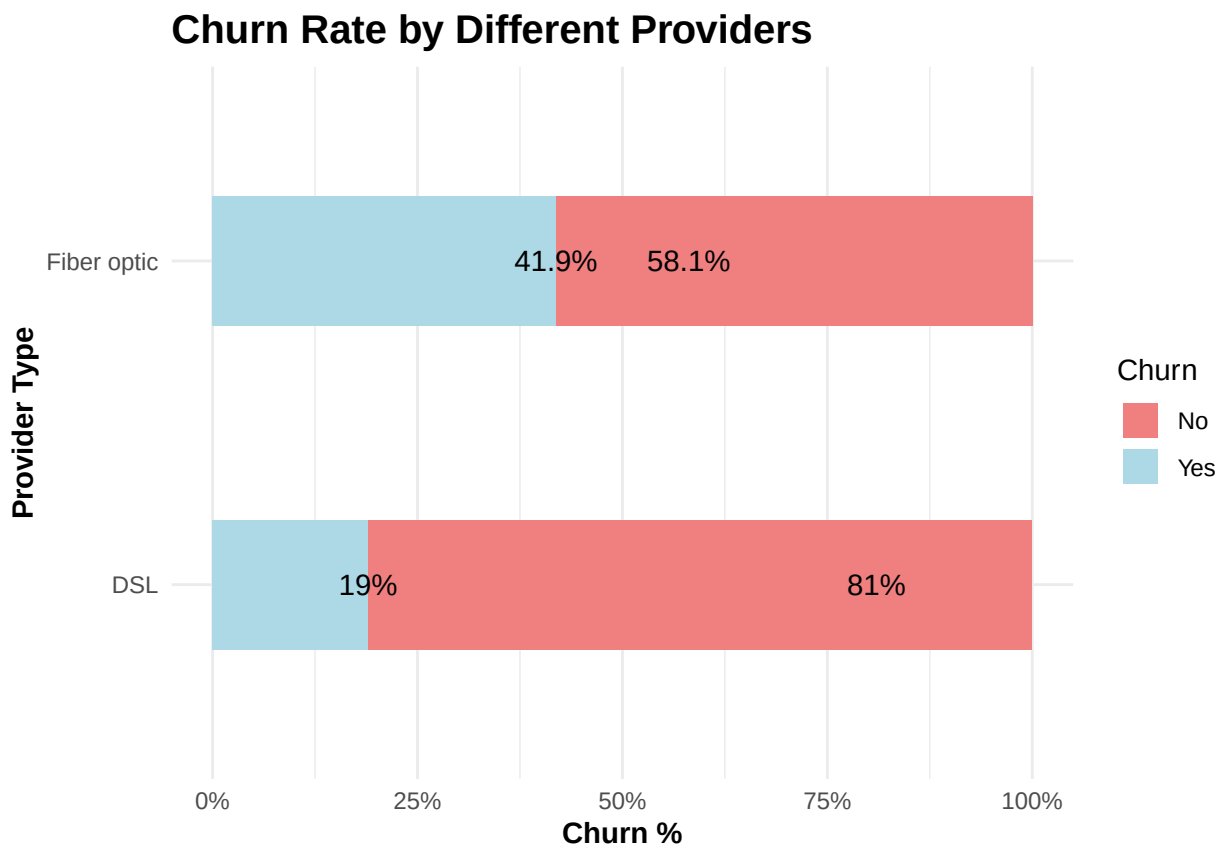
```
(p5 | p6) +
  plot_layout(widths = c(1, 1.3))
```



The churn rate of customers with internet service provider is significantly higher than customers without internet service provider. Due to the significant difference in base size between the two groups (customers with provider-support far exceeds that of customers without), it can be seen from the right picture that the majority of churn comes from the group of customers with internet service provider. Therefore, the subsequent segment focus should put into internet service inside to locate the specific high-risk source.

```
# churn rate under provider-support and different providers
p7 <- df_churn_fe %>%
  filter(has_internet == 'Provider-support') %>%
  count(InternetService, Churn) %>%
  group_by(InternetService) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(
    aes(x = percent,
```

```
        y = InternetService,
        fill = Churn)
    ) +
  geom_col(position = 'fill',
           width = 0.4, linewidth = 1.5) +
  geom_text(
    aes(label = paste0(round(percent*100, 1), '%'))
    , vjust = 0.5) +
  scale_x_continuous(labels = percent) +
  scale_fill_manual(
    values = c(
      'No' = 'lightcoral',
      'Yes' = 'lightblue'
      )
    ) +
  labs(
    title = 'Churn Rate by Different Providers',
    x = 'Churn %',
    y = 'Provider Type',
    fill = 'Churn'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
p7
```



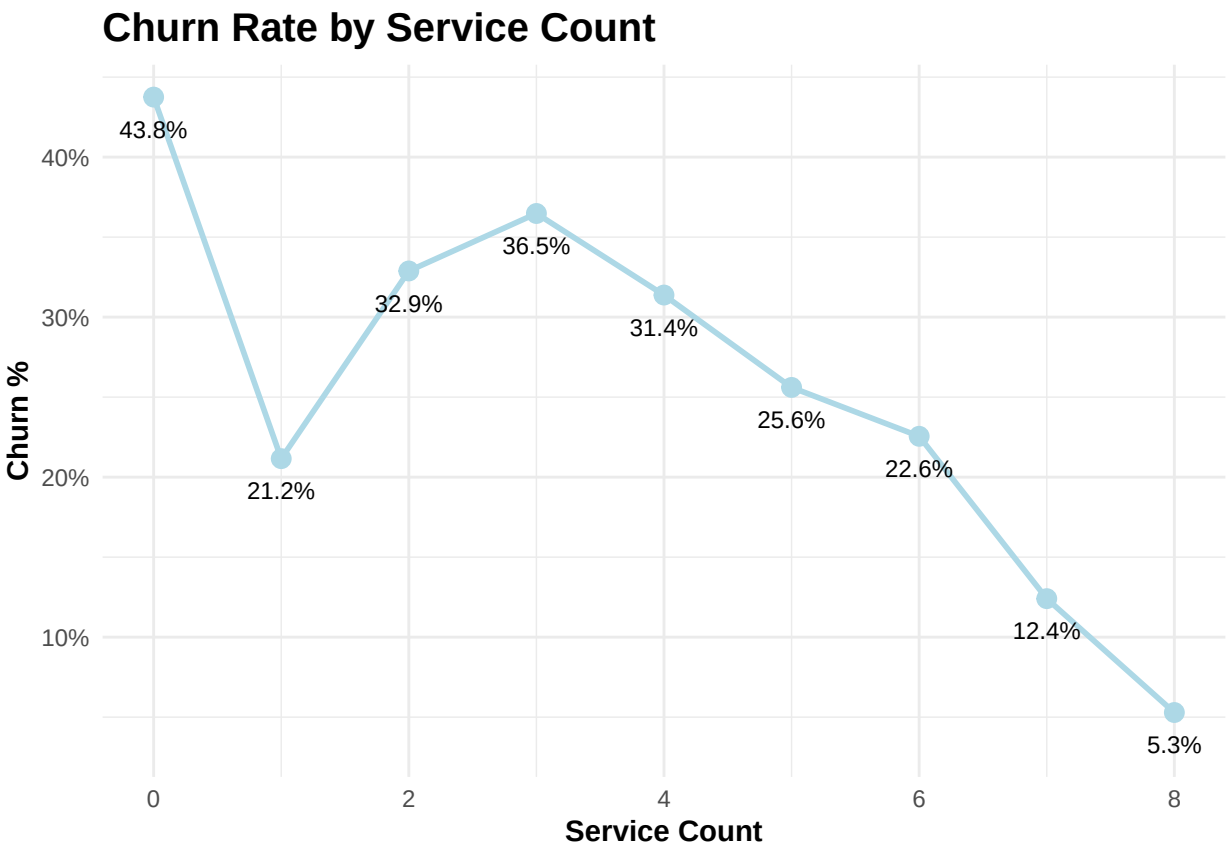
Among customers with internet service, the churn rate between two service providers is different: the churn rate of Fiber optic is significantly higher than DSL (about 41.9% vs 19%). The difference might relate to pricing strategy, service stability, or customer structure (such as family or individual, whether bundling more value-added services).

The result reminders that Fiber optic customer group is high-risk pool. Subsequently, cross-validation can be further conducted in combination with variables such as *charges_bucket* and *services_count* to support more specific operational actions (such as improving service support and optimizing package combinations).

5.4 Bundling & Engagement Proxy

```
p8 <- df_churn_fe %>%
  count(services_count, Churn) %>%
  group_by(services_count) %>%
  mutate(percent = n / sum(n)) %>%
  filter(Churn == 'Yes') %>%
  ggplot(
    aes(x = services_count,
```

```
        y = percent)
    ) +
  geom_line(color = 'lightblue',
            linewidth = 1) +
  geom_point(color = 'lightblue',
             size = 3) +
  geom_text(
    aes(
      label = paste0(round(percent * 100, 1), '%')
      )
    , size = 3
    , nudge_y = -0.02) +
  scale_y_continuous(labels = percent) +
  labs(
    title = 'Churn Rate by Service Count',
    x = 'Service Count',
    y = 'Churn %'
    ) +
  theme_minimal(base_size = 11) +
  theme(
    plot.title = element_text(face = 'bold', size = 15),
    axis.title = element_text(face = 'bold')
    )
p8
```



The overall trend shows: higher service count, lower churn rate, which conforms to the intuition that "the more bundles there are → the higher the conversion cost → the less likely it is to be churned". Especially after service count reaches 6+, the churn rate drops more significantly, indicating a strong correlation between value-added services and retention.

It should be noted that the churn rate of $services\_count = 0$ is abnormally high, which may be related to the small customer size or the special business definition of this population (for example, only a few basic services are retained or customers are about to churn).

In terms of business, a retention strategy direction can use "enhancing service bundling" (for example, designing combination packages or discount incentives for customers with 0-2 services), and combine with high-risk group rules for priority promotion.

5.5 High-risk Segment

A. Candidate high-risk segment rules

Rule A: Non-automatic & New customers

Condition: $auto\_pay$ = "Non-automatic" and $tenure\_bucket$ in (0-6, 7-12).

Explanation: New customers are easier to churn in the early lifecycle. With the addition of non-automatic fee deduction usually represents the unstable payment process or renew, thereby identifying

them as the priority intervention population.

Rule B: Non-automatic & High monthly charges

Condition: *auto_pay* = "Non-automatic" and *charges_bucket* in (Mid-High, High).
Explanation: Churned customers with higher monthly charges will result in higher loss in income. If not adopting automatic fee deduction at the same time, there might appear situations where prices are sensitive or renewal decisions are repeated. This can be as the operation focus among high value and high risk customers.

Rule C: Fiber optic & Low service bundling (<=2)

Condition: *InternetService* = "Fiber optic" and *services_count* <= 2.
Explanation: Fiber optic's customers has relatively higher churn rate. When service subscription is less, customer stickiness and conversion costs will be lower, the risk of churn is further increasing. This combination is used to determine groups with high risk and their stickiness can be enhanced through bundling.

B. Each segment evaluation

```
# calculate overall churn rate
overall_churn_rate <- mean(df_churn_fe$Churn == "Yes")
```

```
# table for rule A
rule_A <- df_churn_fe %>%
  filter(
    auto_pay == 'Non-automatic',
    tenure_bucket %in% c('0-6 months', '7-12 months')
    ) %>%
  summarise(
    segment_size = n(),
    population_share = round(n() / nrow(df_churn_fe), 4),
    churners = sum(Churn == 'Yes'),
    segment_churn_rate = round(churners / n(), 4),
  ) %>%
  mutate(
    rule_name = 'Non-automatic & New customers',
    lift = round(segment_churn_rate / overall_churn_rate, 4)
  )
```

```r
# table for rule B
rule_B <- df_churn_fe %>%
  filter(
    auto_pay == 'Non-automatic',
    charges_bucket %in% c('Mid-High', 'High')
    ) %>%
  summarise(
    segment_size = n(),
    population_share = round(n() / nrow(df_churn_fe), 4),
    churners = sum(Churn == 'Yes'),
    segment_churn_rate = round(churners / n(), 4),
  ) %>%
  mutate(
    rule_name = 'Non-automatic & High month charges',
    lift = round(segment_churn_rate / overall_churn_rate, 4)
  )
```

```r
# table for rule C
rule_C <- df_churn_fe %>%
  filter(
    InternetService == 'Fiber optic',
    services_count <= 2
  ) %>%
  summarise(
    segment_size = n(),
    population_share = round(n() / nrow(df_churn_fe), 4),
    churners = sum(Churn == 'Yes'),
    segment_churn_rate = round(churners / n(), 4)
  ) %>%
  mutate(
    rule_name = 'Fiber optic & Services subscription  2',
    lift = round(segment_churn_rate / overall_churn_rate, 4)
  )
```

```r
# combine all tables
rule_table <- bind_rows(rule_A, rule_B, rule_C) %>%
  relocate(rule_name) %>%
  arrange(desc(lift))
rule_table
```

```
##                                 rule_name segment_size population_share
## 1 Fiber optic & Services subscription   2          664           0.0944
## 2           Non-automatic & New customers         1724           0.2452
## 3      Non-automatic & High month charges         1907           0.2712
##   churners segment_churn_rate   lift
## 1      393             0.5919 2.2270
## 2      853             0.4948 1.8617
## 3      896             0.4698 1.7676
```

The table above evaluates each candidate rule from three aspects:

1. **Population share**: the proportion of this group of people among the total customers.
2. **Segment churn rate**: the internal churn rate within this group of people.
3. **Lift vs overall**: the multiple relative to the overall churn rate, being used for prioritization.

The result displays:

- **Rule C** has highest **lift** (most concentrated risk) but with relatively small group size, which is more suitable for refined high-priority operation/retention.
- **Rule A** focuses on customer size and new customer retention. Its churn rate is significantly higher than overall, which can be as main outbound marketing pool when the budget is limited.
- **Rule B** covers more customers (higher **population_share**) and higher values, which is suitable as a high-value customer protection pool.

There might be overlapping groups of customers in each rule. If used for actual implementation, it can be sorted by *lift* or potential revenue loss, and the final promotion list can be generated after removing duplicates from customers.

Phase 6: Key Insights & Business Recommendations

6.1 Key Insights

Based on grouping and rules filtering in Phase 5, the risk of churn presents a superimposed effect of **payment method friction + early lifecycle + price/service scenario**. Among three candidate high-risk rules, **non-automatic fee deduction** reappears in two large pools, indicating a significant association with churn rate. Meanwhile, **Fiber optic and less bundling services (<=2)** has the highest churn rate (59.19%, lift = 2.23), showing stronger experience/match issues or lack of service stickiness.

6.2 Prioritized Segments

- **P0: Non-automatic & High month charges** (cover 27.12% customers, churn rate 46.98%, lift = 1.77). It has the largest coverage and is given priority as the main battlefield for "reducing the overall number of churned customers".
- **P1: Non-automatic & New customers** (cover 24.52% customers, churn rate 49.48%, lift = 1.86). Early churn is significant and is suitable for rapid improvement through onboarding and payment guidance.
- **P2: Fiber optic & Serivces subscription <= 2** (cover 9.44% customers, churn rate 59.19%, lift = 2.23). It has the highest risk concentration and is recommended as a special governance target for the Fiber optic experience and bundling strategy.

6.3 Recommended Actions

For P0

1. Prioritize enhancing the penetration of automatic fee deduction (such as one-click activation, bill reminders, and incentive mechanisms) to reduce payment friction.
2. Offer more matched package paths (lower fee/add benefits without price increase/convert short-term discounts into long-term value) to high-monthly charge customers, reducing the churn caused by price sensitivity.
3. Provide more clear bill explanation and fee transparency to reduce the impact of charges.

For P1

1. Build 30-day or 60-day of retention reach for new customers (using guidance, frequently asked questions, bill prompts, and striking customer service entry).
2. Set up automatic fee deduction conversion plans for new customers (first-month discounts, credit points, exclusive benefits), using "payment method upgrade" as an early retention handle.

For P2

1. Track the service experience with a focus on the Fiber optic customers (indicators such as failure rate/response rate/satisfaction), locating the root cause of high churn rate and optimize service process.
2. Promote bundling packages for customers with *services_count* <= 2 (such as combination of basic network + security/technical support), reducing churn rate by increasing the switching cost.
3. Further subdivide the Fiber optic customers by tenure/monthly charges, identifying whether the new customers of Fiber optic" is in the higher-risk population and give priority to intervene.

6.4 Measurement

Recommending to track:

- Overall churn rate

- Segmented churn rate of P0/P1/P2
- Adoption rate of automatic fee deduction
- Churn rate of Fiber optic (grouped by *services_count*)
- Change of monthly charges, avoiding "retention improved but revenue decreased"

Phase 7 (Optional): Predictive Baseline

Explanation

1. Purpose:

- Use an interpretable baseline model to validate the direction of key variables determined in Phase 5/6.
- Provide a risk-ranking qualified tools.

2. Setup:

- 70/30 dataset split and stratified sampling.
- Features comes from engineered features in Phase 4.
- Model is logistic regression with dummy encoding (0/1).

3. Performance:

- AUC = 0.83: good discrimination in churned and no churned samples.
- Under default threshold (0.5), confusion matrix displays a certain improvement space for churners' Recall (false negative 292).

4. Interpretation:

- Align with EDA: Early tenure, high monthly charges, non-automatic fee deduction, less service bundling $\rightarrow$ Higher risk of churn.

5. How to use:

- As a priority-ranking model rather than production model.
- Next step: Optimizing thresholds, adding interaction items, attempting decision tree model to improve idetification ability to churners.

Code Implementation

```r
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 1.4.1 --

## v broom        1.0.10     v rsample      1.3.1
## v dials        1.4.2      v tailor       0.1.0
## v infer        1.0.9      v tune         2.0.1
## v modeldata    1.5.1      v workflows    1.3.0
## v parsnip      1.4.0      v workflowsets 1.1.1
## v recipes      1.3.1      v yardstick    1.3.2

## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```r
library(broom)
```

```r
set.seed(42)
```

```r
# choose baseline feature set
model_df <- df_churn_fe %>%
  select(Churn, tenure_bucket, charges_bucket, auto_pay,
         has_internet, services_count) %>%
  drop_na()
```

```r
# split training and testing set (70/30)
split_set <- initial_split(model_df,
                           prop = 0.7,
                           strata = Churn)
train_set <- training(split_set)
test_set <- testing(split_set)

count(train_set, Churn)
```

```
##   Churn    n
```

```
## 1    No 3614
## 2   Yes 1308
```

```
count(test_set, Churn)
```

```
##   Churn    n
## 1    No 1549
## 2   Yes  561
```

```
# convert categorical variables into dummy
rec <- recipe(Churn ~ ., data = train_set) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors()) ## remove zero variance
```

```
# define logistic regression model
log_model <- logistic_reg() %>%
  set_engine('glm') %>%
  set_mode('classification')
```

```
# workflow & fit
wf <- workflow() %>%
  add_recipe(rec) %>%
  add_model(log_model)

fit_log <- wf %>%
  fit(data = train_set)
fit_log
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_dummy()
## * step_zv()
##
## -- Model -----------------------------------------------------------------------
##
```

```
## Call:  stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)
##
## Coefficients:
##                   (Intercept)                 services_count
##                       -1.1092                        -0.2590
##    tenure_bucket_X7.12.months   tenure_bucket_X13.24.months
##                       -0.7622                        -1.1779
##   tenure_bucket_X25.48.months   tenure_bucket_X49.72.months
##                       -1.6301                        -2.4319
##        charges_bucket_Mid.Low       charges_bucket_Mid.High
##                        0.2996                         1.4363
##           charges_bucket_High            auto_pay_Automatic
##                        2.3406                        -0.4770
## has_internet_Provider.support
##                        1.4403
##
## Degrees of Freedom: 4921 Total (i.e. Null);  4911 Residual
## Null Deviance:       5699
## Residual Deviance: 4378  AIC: 4400
```

```
# obtain probability and predicted categories
pred <- predict(fit_log, test_set, type = 'prob') %>%
  bind_cols(
    predict(fit_log, test_set, type = 'class')
    ) %>%
  bind_cols(test_set %>%
              select(Churn))

head(pred)
```

```
## # A tibble: 6 x 4
##   .pred_No .pred_Yes .pred_class Churn
##      <dbl>     <dbl> <fct>       <fct>
## 1    0.351    0.649  Yes         Yes
## 2    0.666    0.334  No          No
## 3    0.910    0.0904 No          No
## 4    0.387    0.613  Yes         No
## 5    0.239    0.761  Yes         No
## 6    0.908    0.0916 No          No
```

```r
# AUC
roc_auc(pred, truth = Churn, .pred_Yes, event_level = "second")
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.831
```

```r
# confusion matrix
conf_mat(pred, truth = Churn, estimate = .pred_class)
```

```
##            Truth
## Prediction   No  Yes
##        No  1401  292
##        Yes  148  269
```

```r
# Precision & Recall & F1
metrics(pred, truth = Churn, estimate = .pred_class)
```

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.791
## 2 kap      binary         0.418
```

```r
fit_log %>%
  extract_fit_parsnip() %>%
  tidy() %>%
  arrange(desc(abs(estimate))) %>%
  head(20)
```

```
## # A tibble: 11 x 5
##    term                          estimate std.error statistic  p.value
##    <chr>                            <dbl>     <dbl>     <dbl>    <dbl>
##  1 tenure_bucket_X49.72.months     -2.43     0.146     -16.7  2.09e-62
##  2 charges_bucket_High              2.34     0.244       9.59 9.11e-22
##  3 tenure_bucket_X25.48.months     -1.63     0.120     -13.5  8.15e-42
##  4 has_internet_Provider.support    1.44     0.220       6.54 6.11e-11
##  5 charges_bucket_Mid.High          1.44     0.213       6.75 1.49e-11
```

```
##  6 tenure_bucket_X13.24.months     -1.18     0.120      -9.84 7.45e-23
##  7 (Intercept)                     -1.11     0.130      -8.56 1.17e-17
##  8 tenure_bucket_X7.12.months      -0.762    0.128      -5.95 2.68e- 9
##  9 auto_pay_Automatic              -0.477    0.0836     -5.70 1.18e- 8
## 10 charges_bucket_Mid.Low           0.300    0.206       1.45 1.46e- 1
## 11 services_count                  -0.259    0.0354     -7.31 2.67e-13
```

```r
# disconnect
dbDisconnect(con)
```