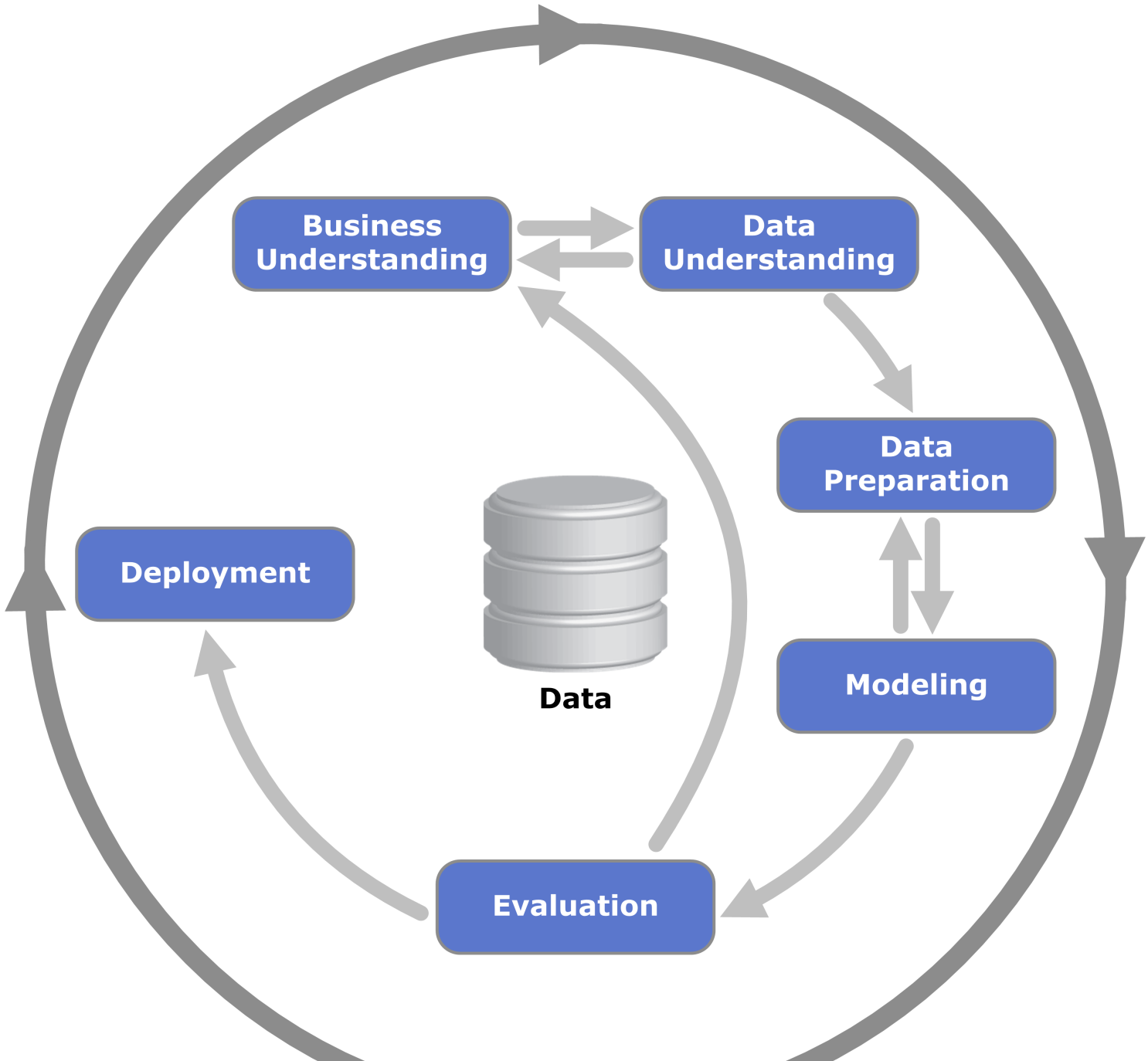
VPS Customer churn prediction

JG Pardyak

18-8-2021

Presentantion outline



Business understanding

Company X sells a Virtual Private Server (VPS) as a service. The company wants to know which customers intend to leave VPS so they can devise an appropriate customer re-engagement strategy before it's too late.

Data understanding

```
library(tidyverse)
library(tidymodels)
vps <- read_csv("./data/vps_churn_data.txt") %>%
  mutate(is_churn = factor(ifelse(is_churn == 0,
                                  "No", "Yes")))

vps %>% dim()
```

```
## [1] 283  23
```

In this work we use *R* and *tidy*- libraries. All commands are visible to facilitate the verification of the presentation. We see that the dataset is composed of 283 observations described with 23 variables.

Data understanding cont.

The two classes (is_churn = Yes and is_churn = No) are almost equally distributed.

```
vps %>%
  count(is_churn) %>%
  mutate(prop = n/sum(n))
```

```
## # A tibble: 2 x 3
##   is_churn     n  prop
##   <fct>    <int> <dbl>
## 1 No         148 0.523
## 2 Yes        135 0.477
```

Data understanding cont.

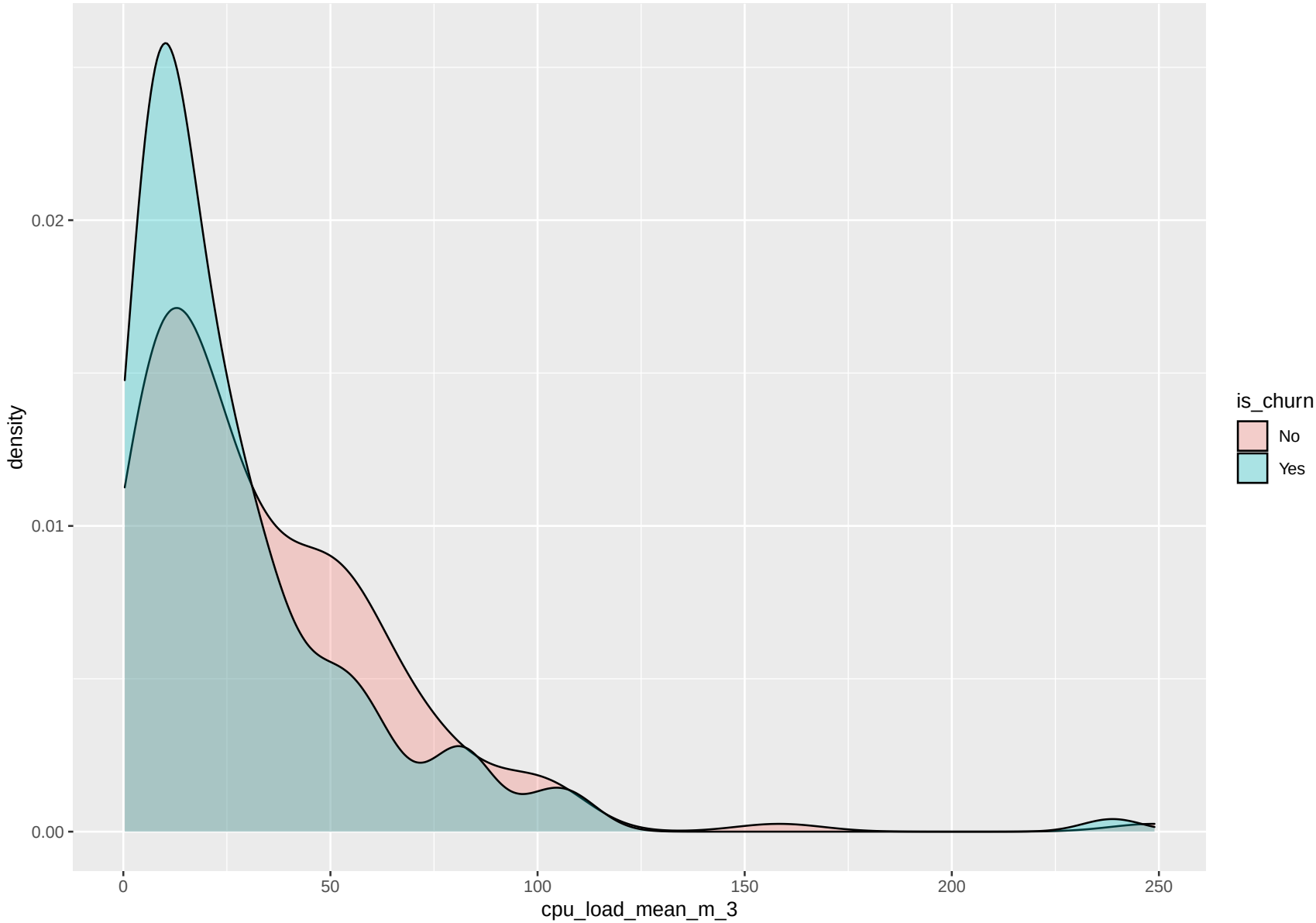
The dataset is complete - there are no missing values we have to deal with.

```
vps %>% is.na() %>% colSums() %>% head(6)
```

```
##                              id          cpu_load_mean_m_3
##                               0                          0
##  disk_octets_read_mean_m_3 disk_octets_write_mean_m_3
##                               0                          0
##     disk_ops_read_mean_m_3    disk_ops_write_mean_m_3
##                               0                          0
```

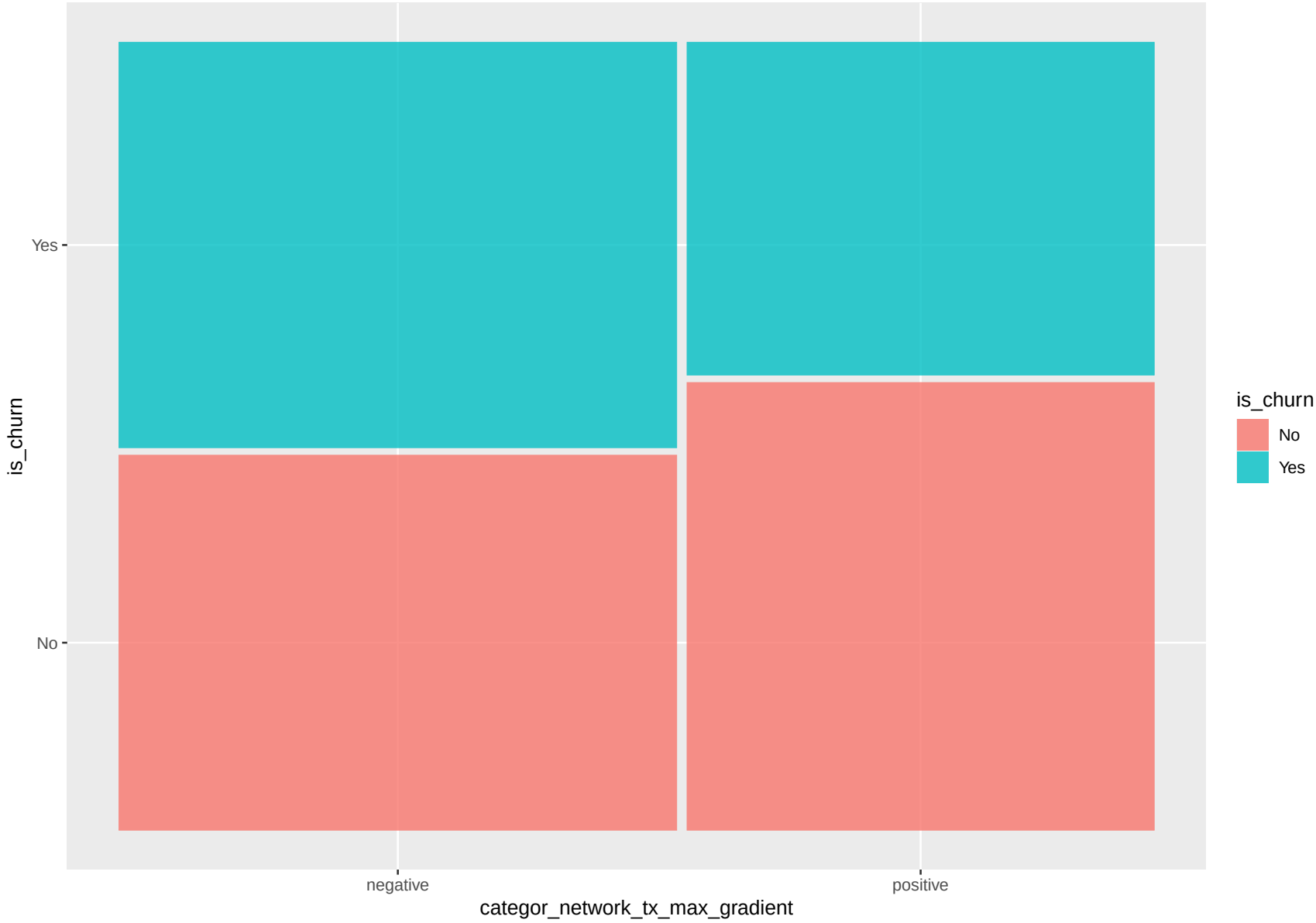
Data understanding cont.

We check whether first variable can serve as a “good” predictor.



Data understanding cont.

We check whether last variable can serve as a “good” predictor.



Data preparation

We split our data into training and test datasets.

```
set.seed(1234)
vps_split <- initial_split(vps, prop = 0.6,
                                strata = is_churn)
vps_split
```

```
## <Analysis/Assess/Total>
## <169/114/283>
```

Data preparation cont.

We check the observations used for training.

```
## Rows: 169
## Columns: 23
## $ id                                    <dbl> 102, 104, 1
## $ cpu_load_mean_m_3                     <dbl> 248.935914
## $ disk_octets_read_mean_m_3             <dbl> 19.3715277
## $ disk_octets_write_mean_m_3            <dbl> 31.9710273
## $ disk_ops_read_mean_m_3                <dbl> 213.160322
## $ disk_ops_write_mean_m_3               <dbl> 194.558279
## $ network_rx_mean_m_3                   <dbl> 1.72831136
## $ network_tx_mean_m_3                   <dbl> 1.35480347
## $ cpu_load_monthly_mean_delta           <dbl> 9.67491398
## $ network_tx_monthly_mean_delta         <dbl> -0.190598
## $ network_rx_monthly_mean_delta         <dbl> -0.331511
## $ disk_ops_read_monthly_mean_delta      <dbl> -22.40323
## $ disk_ops_write_monthly_mean_delta     <dbl> -26.08083
## $ disk_octets_write_monthly_mean_delta <dbl> -2.343395
## $ disk_octets_read_monthly_mean_delta  <dbl> -0.535923
```

Data preparation cont.

We write recipe to prepare our data for training. Steps are described in comments.

```
vps_recipe <- training(vps_split) %>% # on which data
  recipe(is_churn ~.) %>% # training formula
  step_rm(id) %>% # step remove id column
  # remove variables highly correlated with other vars
  step_corr(all_predictors()) %>%
  # make vars to be of mean zero
  step_center(all_predictors(), -all_outcomes()) %>%
  # make vars to be standard dev of 1
  step_scale(all_predictors(), -all_outcomes()) %>%
  prep() # execute transformations
```

Data preparation cont.

We use previously written recipe to prepare *training* data.

```
vps_training <- juice(vps_recipe)
vps_training %>% select(1:10) %>% glimpse()
```

```
## Rows: 169
## Columns: 10
## $ cpu_load_mean_m_3                 <dbl> 6.9540144, 1.7
## $ disk_octets_read_mean_m_3         <dbl> 0.7810093, 0.5
## $ disk_octets_write_mean_m_3        <dbl> 2.65330241, 1.
## $ disk_ops_read_mean_m_3            <dbl> 0.02212873, 0.
## $ disk_ops_write_mean_m_3           <dbl> 0.66829422, -0
## $ network_rx_mean_m_3               <dbl> -0.10276295, -
## $ network_tx_mean_m_3               <dbl> -0.190306768,
## $ network_tx_monthly_mean_delta     <dbl> -0.016170786,
## $ disk_ops_read_monthly_mean_delta  <dbl> -0.1108472650,
## $ disk_ops_write_monthly_mean_delta <dbl> -0.1447599445,
```

Data preparation cont.

We use previously written recipe to prepare *test* data.

```
vps_testing <- vps_recipe %>%
  bake(testing(vps_split))
vps_testing %>% select(1:10) %>% glimpse()
```

```
## Rows: 114
## Columns: 10
## $ cpu_load_mean_m_3                  <dbl> -0.608674397,
## $ disk_octets_read_mean_m_3          <dbl> 1.92677245, -0
## $ disk_octets_write_mean_m_3         <dbl> -0.36157310, -
## $ disk_ops_read_mean_m_3             <dbl> 0.328112376, -
## $ disk_ops_write_mean_m_3            <dbl> -0.29893573, -
## $ network_rx_mean_m_3                <dbl> -0.1932091, -0
## $ network_tx_mean_m_3                <dbl> -0.2665257913,
## $ network_tx_monthly_mean_delta      <dbl> 0.0849835086,
## $ disk_ops_read_monthly_mean_delta   <dbl> 1.85174047, 0.
## $ disk_ops_write_monthly_mean_delta <dbl> 0.09257490, 0.
```

Modeling & Evaluation

Further we will interlace **Modeling** and **Evaluation** steps for selected algorithms. Models are trained on previously selected sample and tested on another. Predicting power of each model is measured with *Accuracy* and *Area under curve* measures. We start from *Null model* - assumption that no one will churn.

```
null_model <- null_model(mode = "classification") %>%
  set_engine("parsnip") %>%
  fit(is_churn ~ ., data = vps_training)
```

Modeling & Evaluation - null_model

```
## # A tibble: 2 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.526
## 2 kap      binary         0
```

Modeling & Evaluation - null_model

Modeling & Evaluation - null_model