Why you need logistic regression

INTRODUCTION TO REGRESSION IN R

Richie Cotton

Learning Solutions Architect at DataCamp

Bank churn dataset

has_churned	time_since_first_purchase	time_since_last_purchase
0	0.3993247	-0.5158691
1	-0.4297957	0.6780654
0	3.7383122	0.4082544
0	0.6032289	-0.6990435
...	...	...
response	*length of relationship*	*recency of activity*

[1] https://www.rdocumentation.org/packages/bayesQR/topics/Churn

Churn vs. recency: a linear model

```
mdl_churn_vs_recency_lm <- lm(has_churned ~ time_since_last_purchase, data = churn)
```

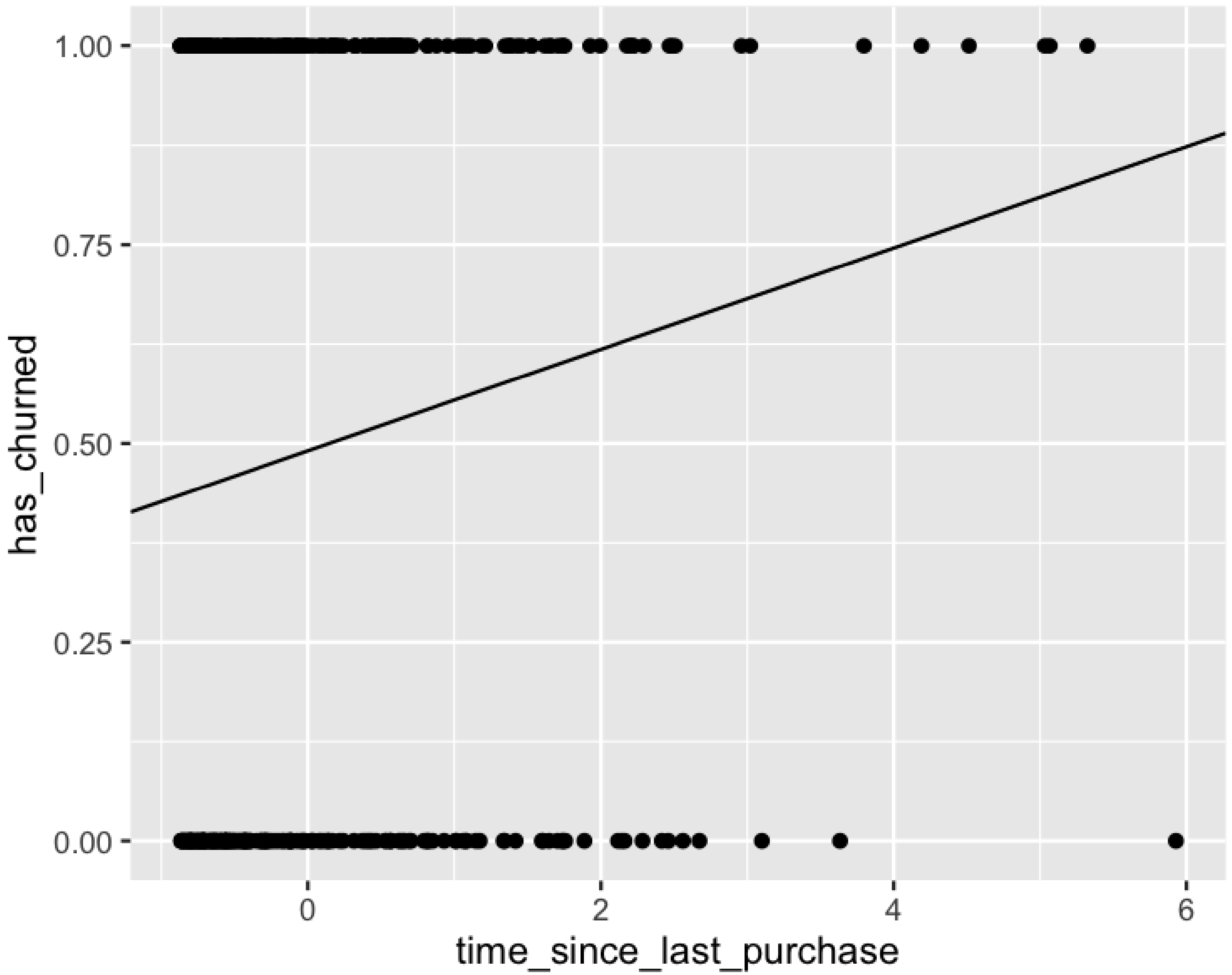
```
Call:
lm(formula = has_churned ~ time_since_last_purchase, data = churn)

Coefficients:
             (Intercept)  time_since_last_purchase
                 0.49078                   0.06378
```

```
coeffs <- coefficients(mdl_churn_vs_recency_lm)
intercept <- coeffs[1]
slope <- coeffs[2]
```

Visualizing the linear model

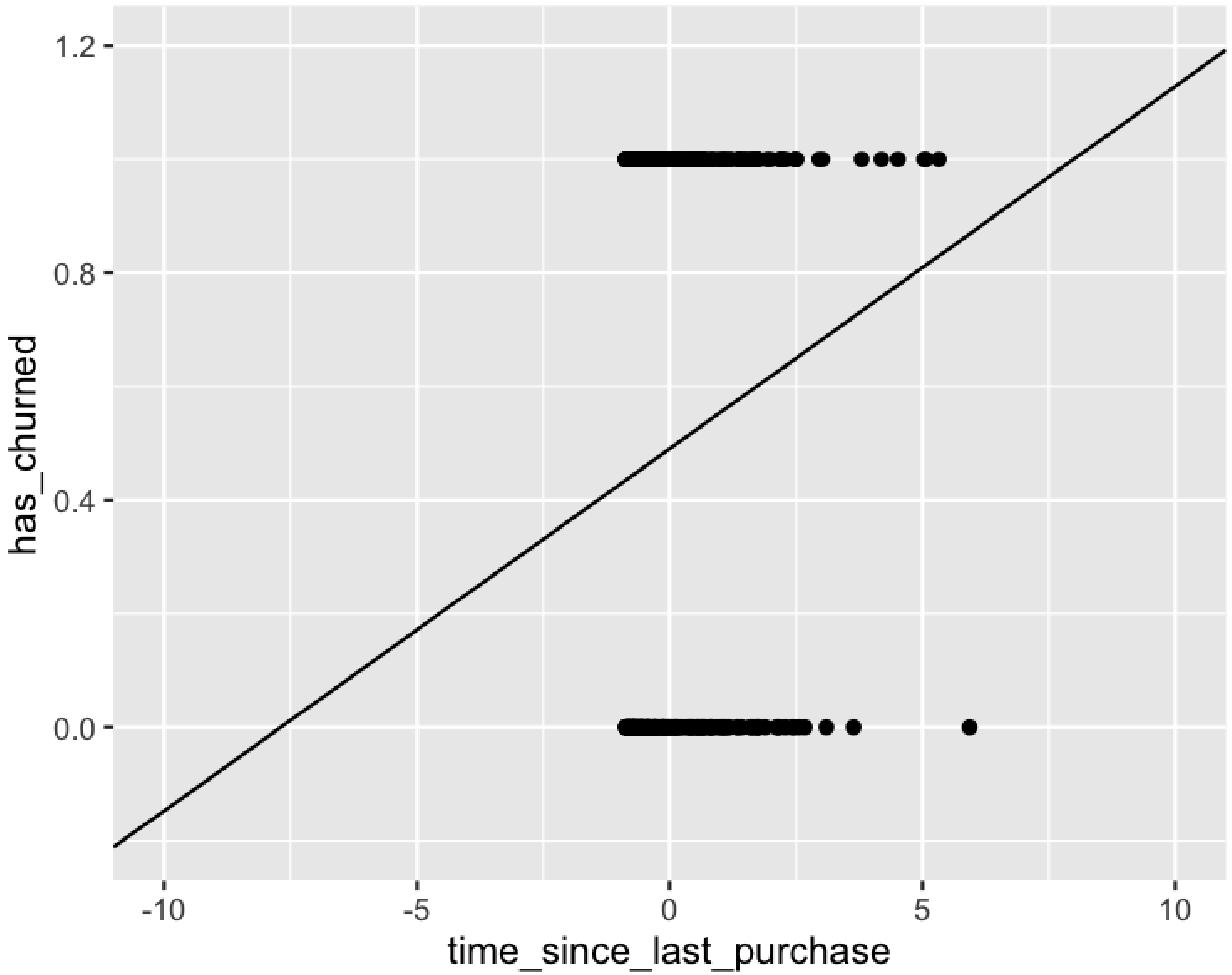
```
ggplot(
  churn,
  aes(time_since_last_purchase, has_churned)
) +
  geom_point() +
  geom_abline(intercept = intercept, slope = slope)
```

Predictions are probabilities of churn, not amounts of churn.



Zooming out

```
ggplot(
  churn,
  aes(days_since_last_purchase, has_churned)
) +
  geom_point() +
  geom_abline(intercept = intercept, slope = slope) +
  xlim(-10, 10) +
  ylim(-0.2, 1.2)
```



What is logistic regression?

- Another type of generalized linear model.
- Used when the response variable is logical.
- The responses follow logistic (S-shaped) curve.

Linear regression using glm()

```
glm(has_churned ~ time_since_last_purchase, data = churn, family = gaussian)
```

```
Call:  glm(formula = has_churned ~ time_since_last_purchase, family = gaussian,
    data = churn)

Coefficients:
             (Intercept)  time_since_last_purchase
                 0.49078                   0.06378

Degrees of Freedom: 399 Total (i.e. Null);  398 Residual
Null Deviance:        100
Residual Deviance: 98.02     AIC: 578.7
```

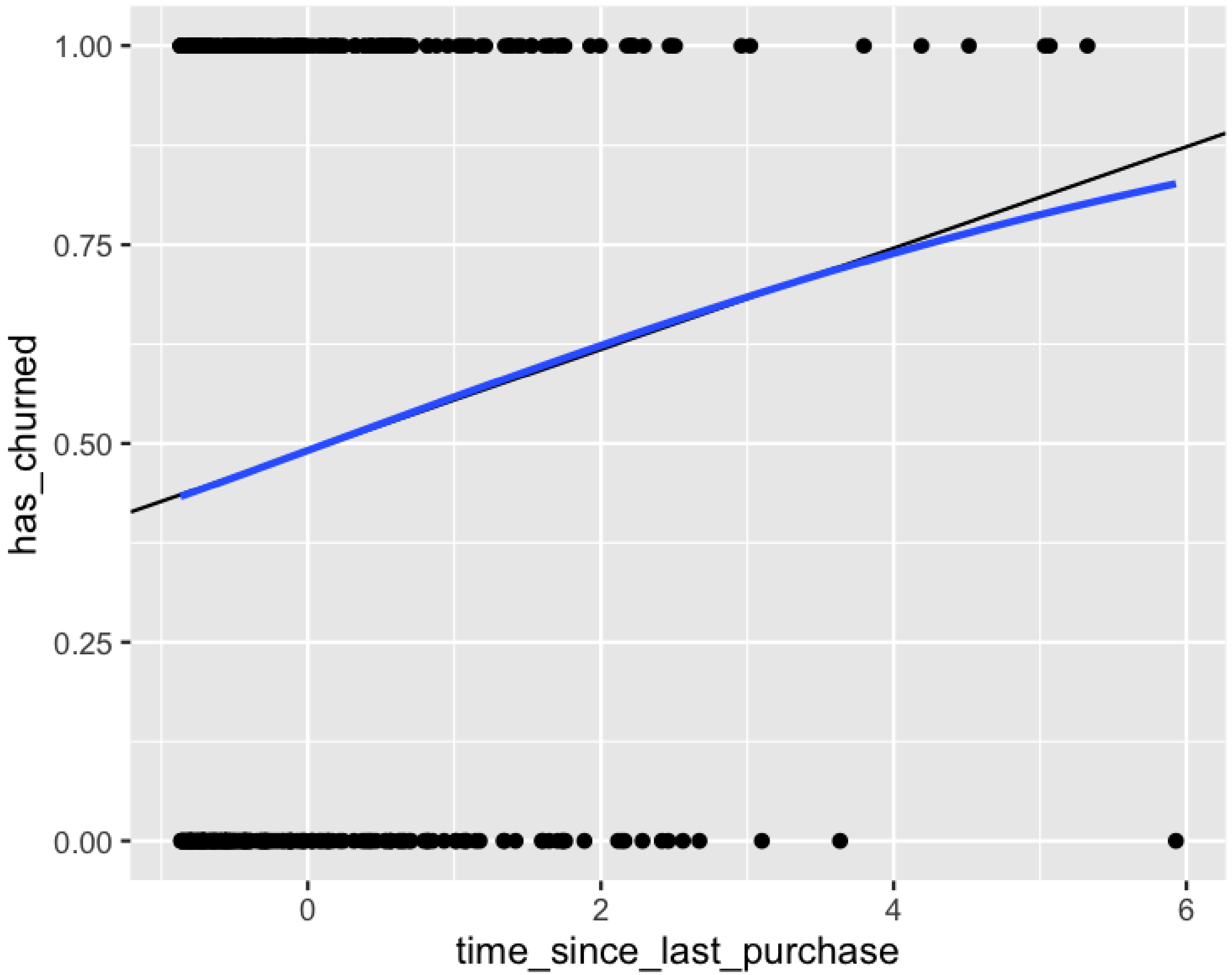
Logistic regression: glm() with binomial family

```
mdl_recency_glm <- glm(has_churned ~ time_since_last_purchase, data = churn, family = binomial)
```

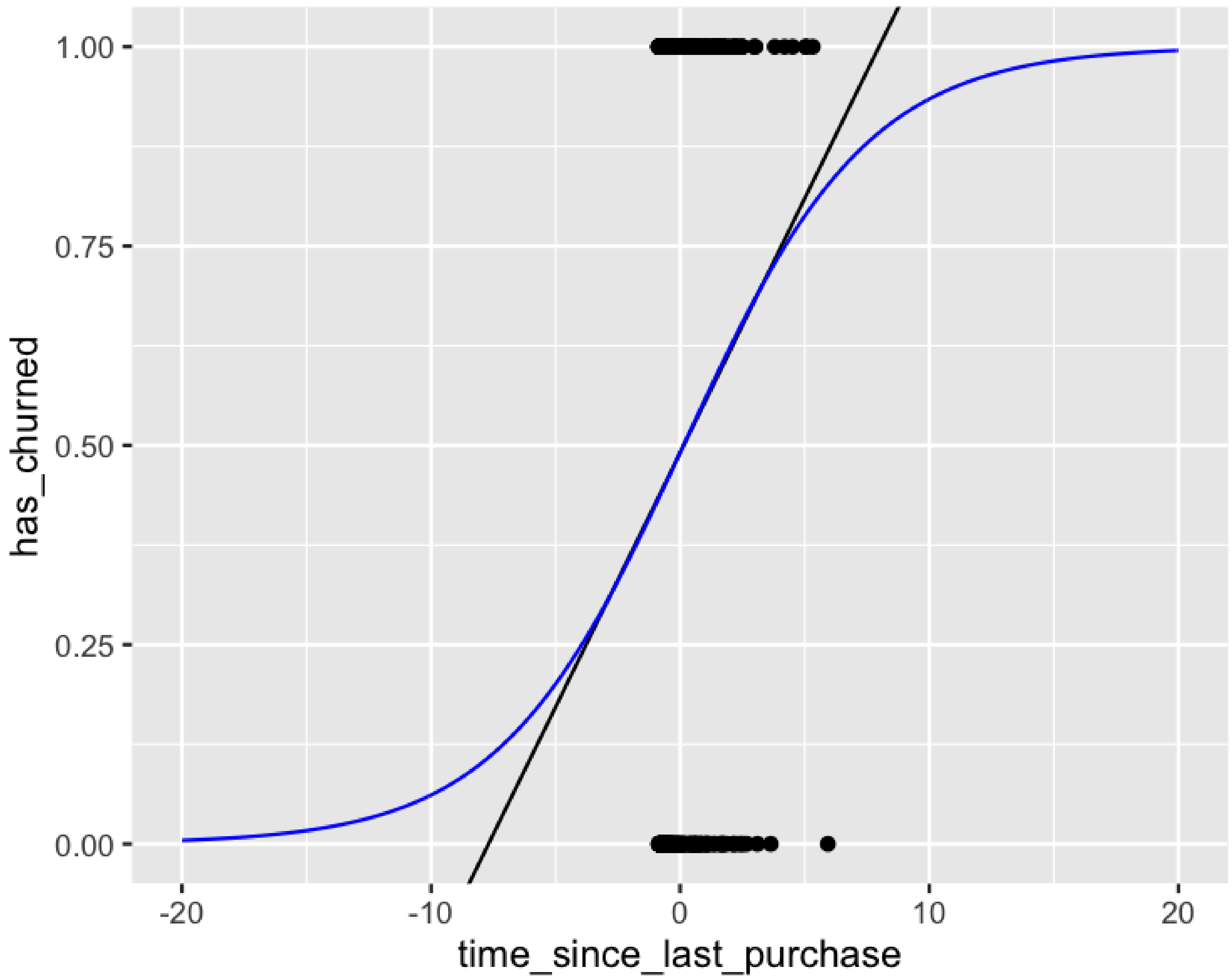
```
Call:  glm(formula = has_churned ~ time_since_last_purchase, family = binomial,
    data = churn)

Coefficients:
             (Intercept)  time_since_last_purchase
                -0.03502                   0.26921

Degrees of Freedom: 399 Total (i.e. Null);  398 Residual
Null Deviance:        554.5
Residual Deviance: 546.4     AIC: 550.4
```

Visualizing the logistic model

```
ggplot(
  churn,
  aes(time_since_last_purchase, has_churned)
) +
  geom_point() +
  geom_abline(
    intercept = intercept, slope = slope
  ) +
  geom_smooth(
    method = "glm",
    se = FALSE,
    method.args = list(family = binomial)
  )
```



Zooming out



1.00
0.75
0.50
0.25
0.00
has_churned
-20
-10
0
10
20
time_since_last_purchase

Let's practice!

INTRODUCTION TO REGRESSION IN R

Predictions and odds ratios

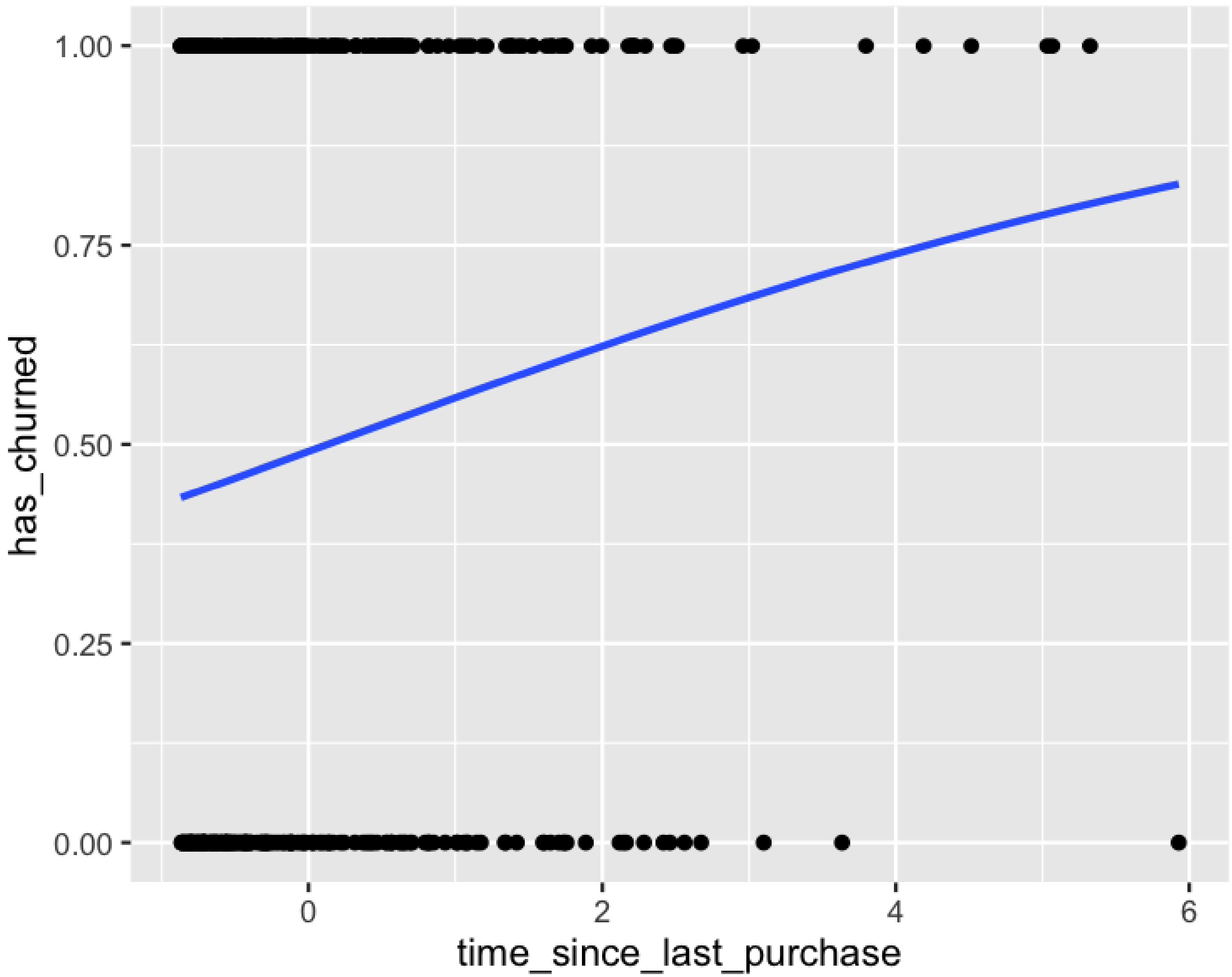
INTRODUCTION TO REGRESSION IN R

Richie Cotton

Learning Solutions Architect at DataCamp

The ggplot predictions

```
plt_churn_vs_recency_base <- ggplot(
  churn,
  aes(time_since_last_purchase, has_churned)
) +
  geom_point() +
  geom_smooth(
    method = "glm",
    se = FALSE,
    method.args = list(family = binomial)
  )
```



Making predictions

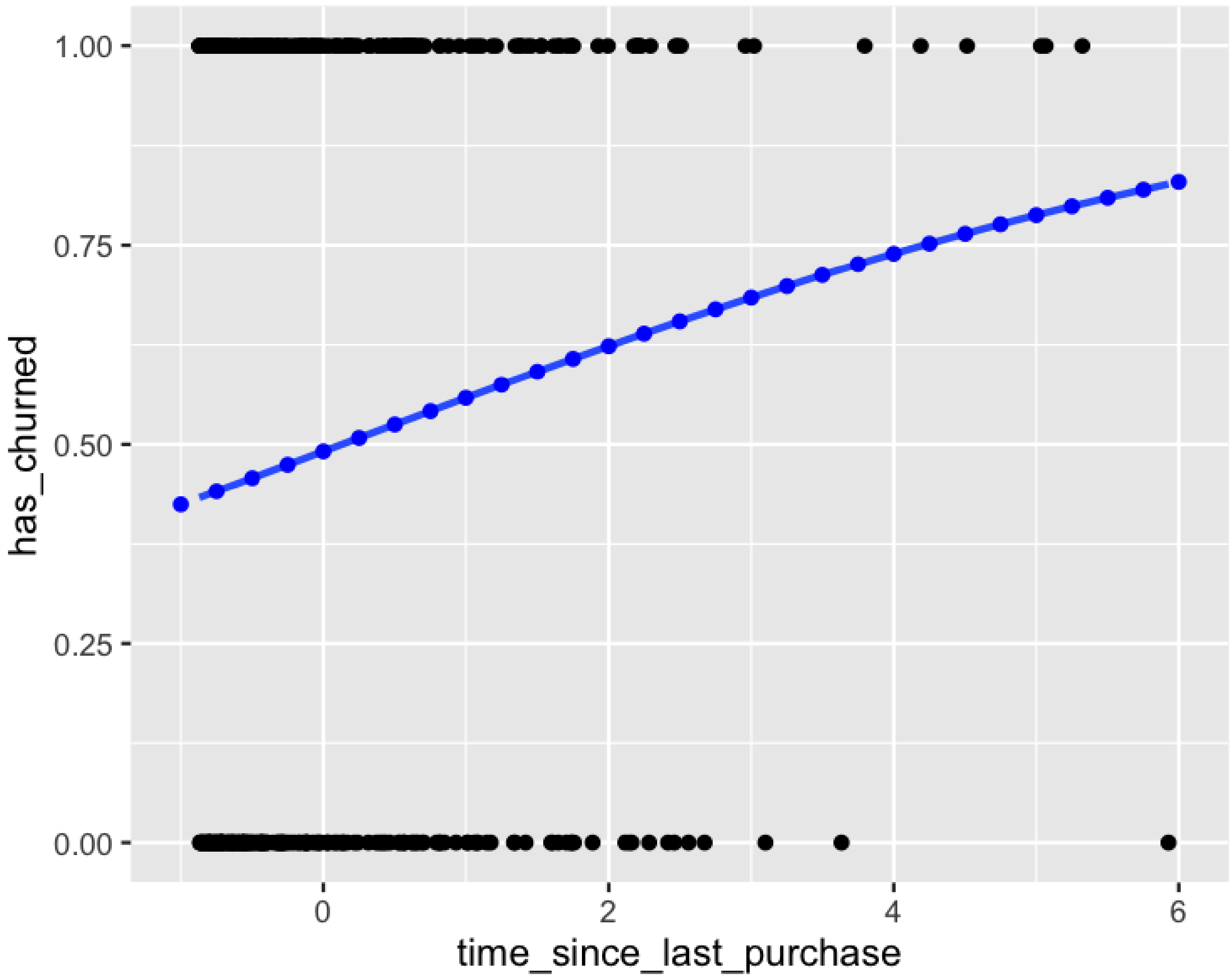
```
mdl_recency <- glm(
  has_churned ~ time_since_last_purchase, data = churn, family = "binomial"
)
```

```
explanatory_data <- tibble(
  time_since_last_purchase = seq(-1, 6, 0.25)
)
```

```
prediction_data <- explanatory_data %>%
  mutate(
    has_churned = predict(mdl_recency, explanatory_data, type = "response")
  )
```

Adding point predictions

```
plt_churn_vs_recency_base +
  geom_point(
    data = prediction_data,
    color = "blue"
  )
```

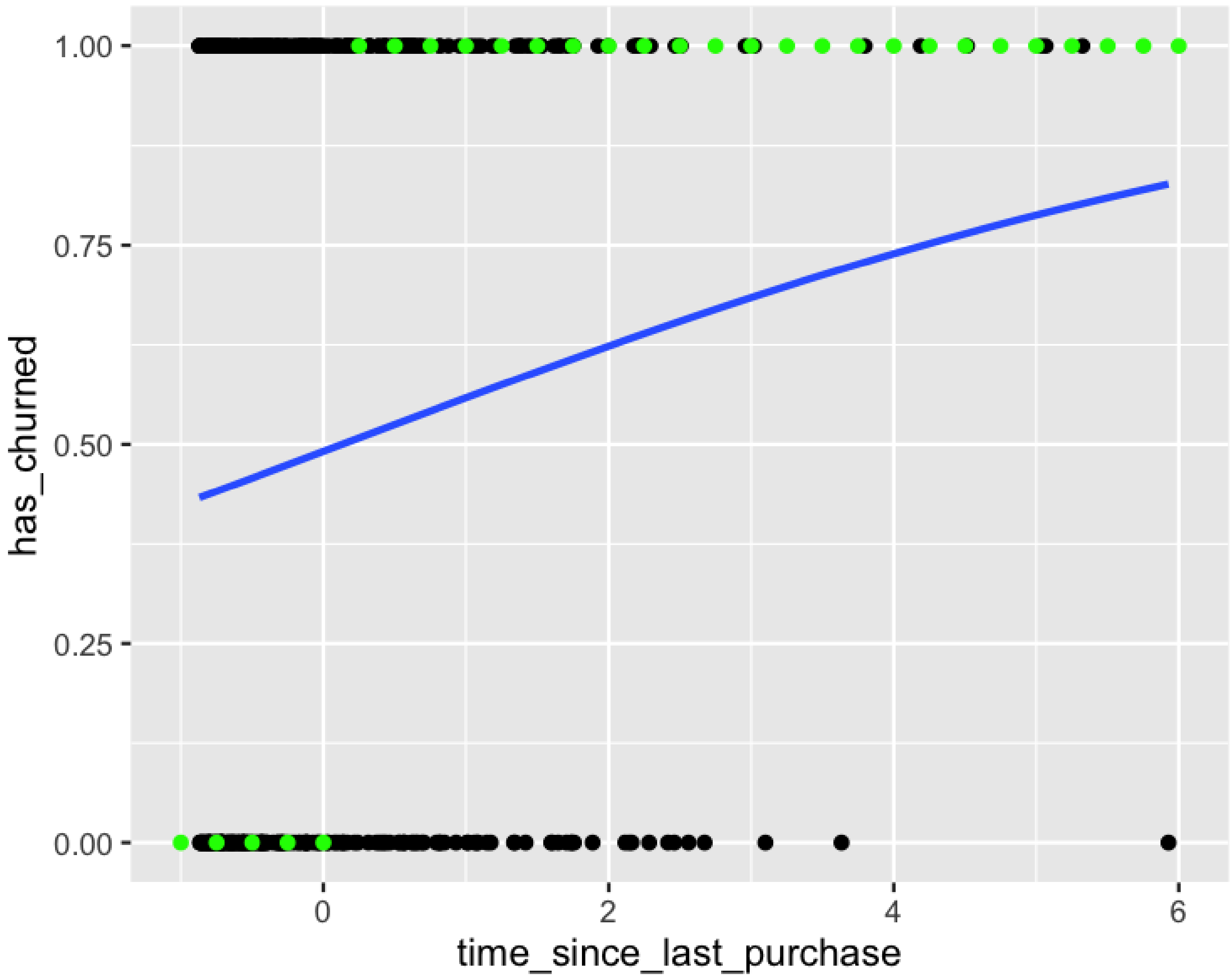


Getting the most likely outcome

```
prediction_data <- explanatory_data %>%
  mutate(
    has_churned = predict(mdl_recency, explanatory_data, type = "response"),
    most_likely_outcome = round(has_churned)
  )
```

Visualizing most likely outcome

```
plt_churn_vs_recency_base +
  geom_point(
    aes(y = most_likely_outcome),
    data = prediction_data,
    color = "green"
  )
```

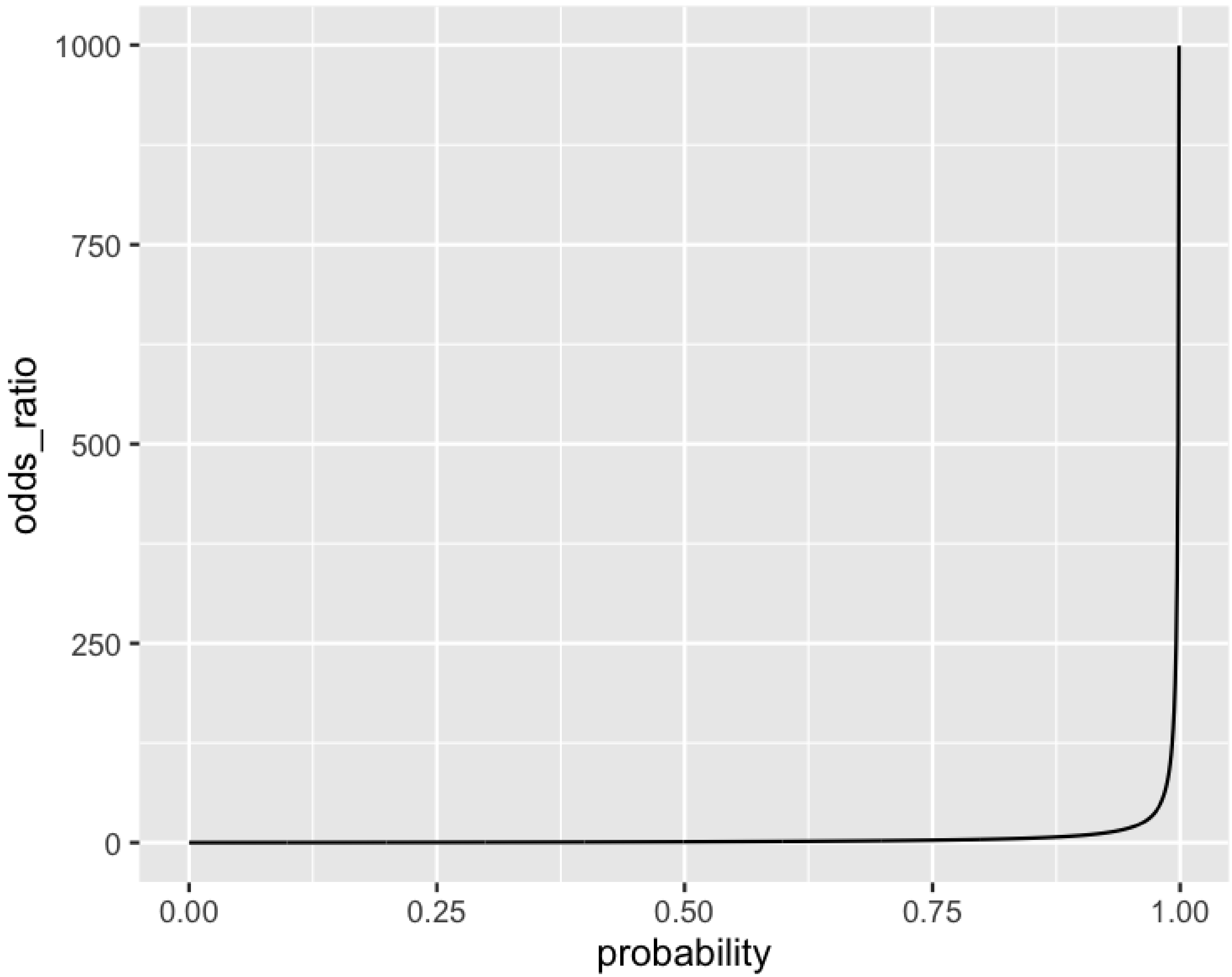


Odds ratios

Odds ratio is the probability of something happening divided by the probability that it doesn't.

$$odds\_ratio = \frac{probability}{(1 - probability)}$$

$$odds\_ratio = \frac{0.25}{(1 - 0.25)} = \frac{1}{3}$$

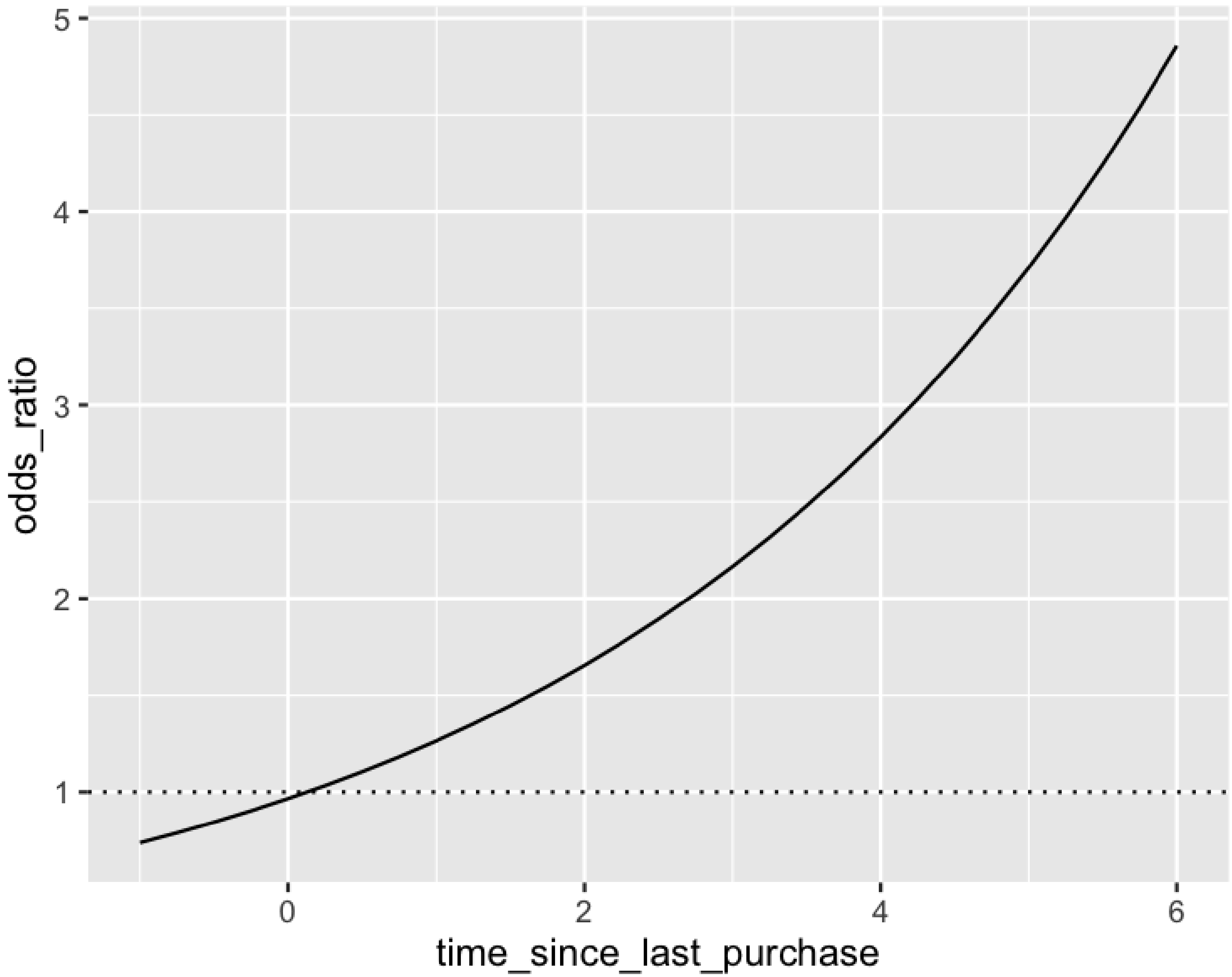


Calculating odds ratio

```
prediction_data <- explanatory_data %>%
  mutate(
    has_churned = predict(mdl_recency, explanatory_data, type = "response"),
    most_likely_response = round(has_churned),
    odds_ratio = has_churned / (1 - has_churned)
  )
```

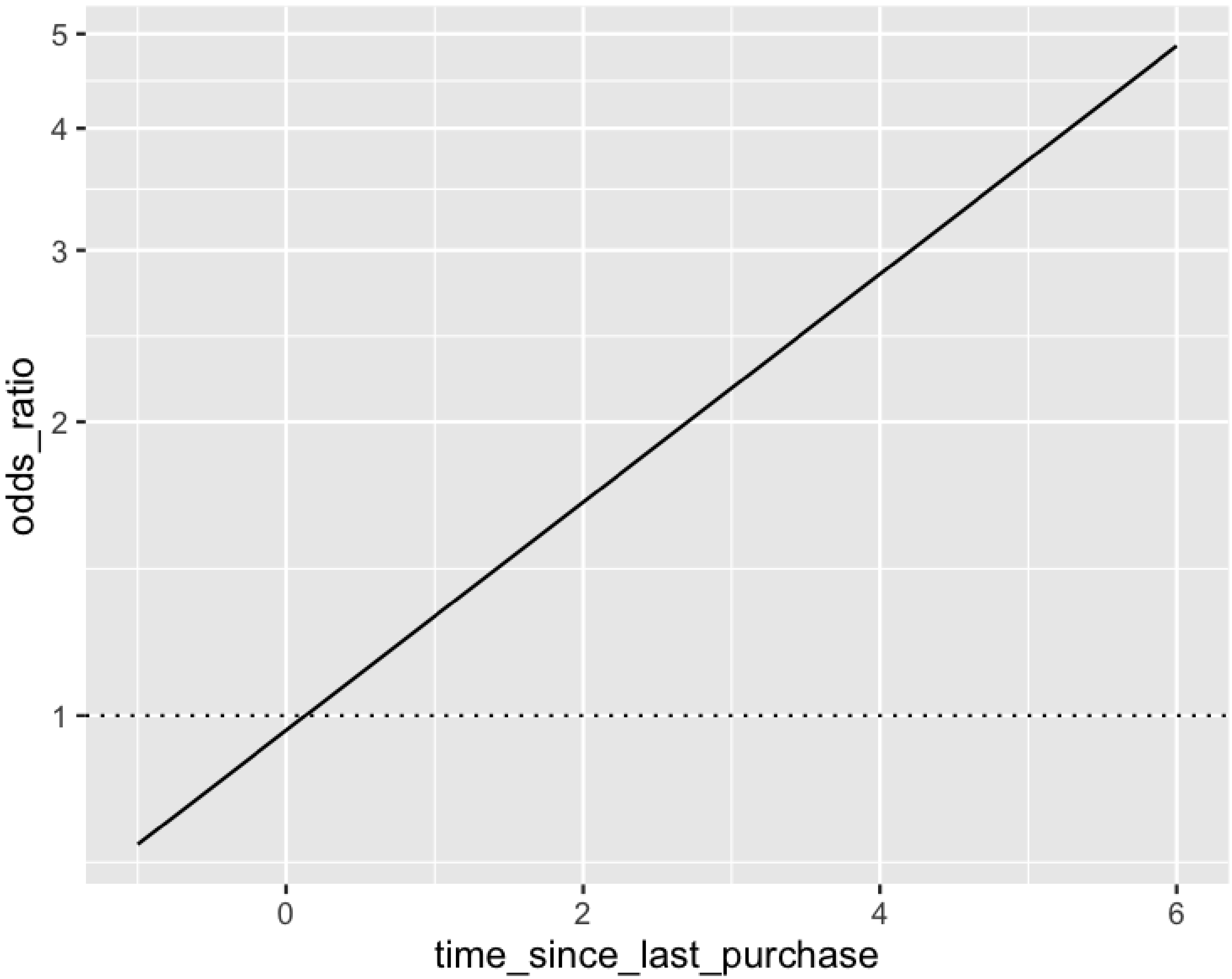
Visualizing odds ratio

```
ggplot(
  prediction_data,
  aes(time_since_last_purchase, odds_ratio)
) +
  geom_line() +
  geom_hline(yintercept = 1, linetype = "dotted")
```



Visualizing log odds ratio

```
ggplot(
  prediction_data,
  aes(time_since_last_purchase, odds_ratio)
) +
  geom_line() +
  geom_hline(yintercept = 1, linetype = "dotted") +
  scale_y_log10()
```



Calculating log odds ratio

```
prediction_data <- explanatory_data %>%
  mutate(
    has_churned = predict(mdl_recency, explanatory_data, type = "response"),
    most_likely_response = round(has_churned),
    odds_ratio = has_churned / (1 - has_churned),
    log_odds_ratio = log(odds_ratio),
    log_odds_ratio2 = predict(mdl_recency, explanatory_data)
  )
```

All predictions together

tm_snc_lst_prch	has_churned	most_lkly_rspns	odds_ratio	log_odds_ratio	log_odds_
0	0.491	0	0.966	-0.035	-
2	0.623	1	1.654	0.503	
4	0.739	1	2.834	1.042	
6	0.829	1	4.856	1.580	
...	...	...	...	...	

Comparing scales

Scale	Are values easy to interpret?	Are changes easy to interpret?	Is precise?
Probability	✔	✘	✔
Most likely outcome	✔✔	✔	✘
Odds ratio	✔	✘	✔
Log odds ratio	✘	✔	✔

Let's practice!

INTRODUCTION TO REGRESSION IN R

Quantifying logistic regression fit

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Learning Solutions Architect at DataCamp

The four outcomes

	actual false	actual true
predicted false	correct	false negative
predicted true	false positive	correct

Confusion matrix: counts of outcomes

```
mdl_recency <- glm(has_churned ~ time_since_last_purchase, data = churn, family = "binomial")
```

```
actual_response <- churn$has_churned
```

```
predicted_response <- round(fitted(mdl_recency))
```

```
outcomes <- table(predicted_response, actual_response)
```

```
                  actual_response
predicted_response   0   1
                 0 141 111
                 1  59  89
```

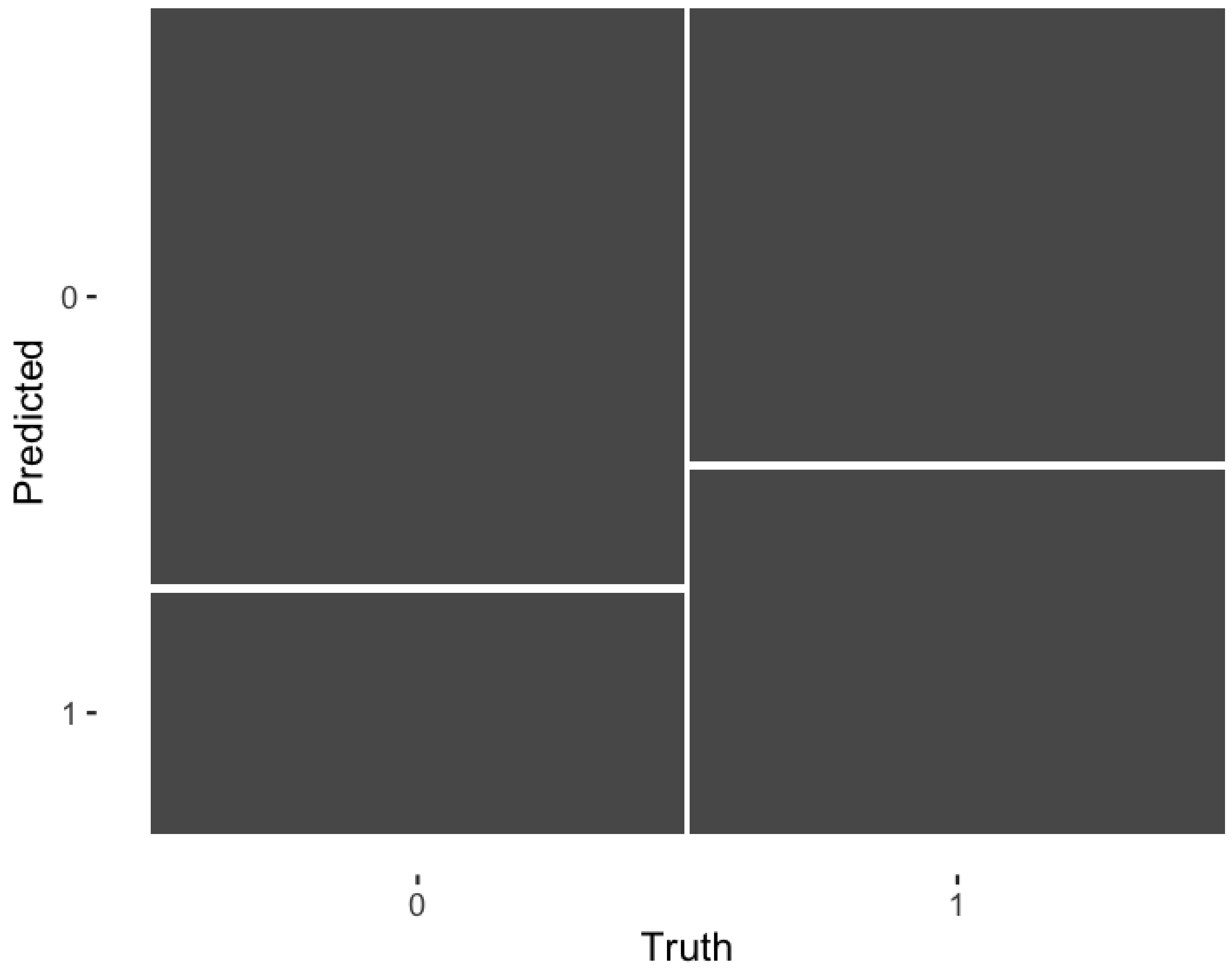
Visualizing the confusion matrix: mosaic plot

```
library(ggplot2)
library(yardstick)
```

```
confusion <- conf_mat(outcomes)
```

```
                 actual_response
predicted_response   0   1
                 0 141 111
                 1  59  89
```

```
autoplot(confusion)
```



Performance metrics

```
summary(confusion, event_level = "second")
```

```
# A tibble: 13 x 3
   .metric              .estimator .estimate
   <chr>                <chr>          <dbl>
 1 accuracy             binary         0.575
 2 kap                  binary         0.150
 3 sens                 binary         0.445
 4 spec                 binary         0.705
 5 ppv                  binary         0.601
 6 npv                  binary         0.560
 7 mcc                  binary         0.155
 8 j_index              binary         0.150
 9 bal_accuracy         binary         0.575
10 detection_prevalence binary         0.37
11 precision            binary         0.601
12 recall               binary         0.445
13 f_meas               binary         0.511
```

Accuracy

```r
summary(confusion) %>%
  slice(1)
```

```
# A tibble: 3 x 3
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.575
```

Accuracy is the proportion of correct predictions.

$$accuracy = \frac{TN + TP}{TN + FN + FP + TP}$$

```r
confusion
```

```
                  actual_response
predicted_response   0   1
                 0 141 111
                 1  59  89
```

```r
(141 + 89) / (141 + 111 + 59 + 89)
```

```
0.575
```

Sensitivity

```
summary(confusion) %>%
  slice(3)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 sens    binary         0.445
```

Sensitivity is the proportion of true positives.

$$sensitivity = \frac{TP}{FN + TP}$$

```
confusion
```

```
                  actual_response
predicted_response   0   1
                 0 141 111
                 1  59  89
```

```
89 / (111 + 89)
```

```
0.445
```

Specificity

```r
summary(confusion) %>%
  slice(4)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 spec    binary         0.705
```

Specificity is the proportion of true negatives.

$$specificity = \frac{TN}{TN + FP}$$

```r
confusion
```

```
                  actual_response
predicted_response   0   1
                 0 141 111
                 1  59  89
```

```r
141 / (141 + 59)
```

```
0.705
```

Let's practice!

INTRODUCTION TO REGRESSION IN R

Congratulations

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Learning Solutions Architect at DataCamp

You learned things

Chapter 1

- Fit a simple linear regression
- Interpret coefficients

Chapter 2

- Make predictions
- Regression to the mean
- Transforming variables

Chapter 3

- Quantifying model fit
- Outlier, leverage, and influence

Chapter 4

- Fit a simple logistic regression
- Make predictions
- Get performance from confusion matrix

Multiple explanatory variables

Multiple and Logistic Regression in R

Unlocking advanced skills

- Modeling with Data in the Tidyverse
- Generalized Linear Models in R
- Machine Learning with caret in R
- Bayesian Regression Modeling with rstanarm

Regression is important everywhere

- Credit Risk Modeling in R
- Building Response Models in R
- Human Resource Analytics, Exploring Employee Data in R
- Predictive Analytics Using Networked Data in R

Let's practice!

INTRODUCTION TO REGRESSION IN R