Model assessment and selection

MODELING WITH DATA IN THE TIDYVERSE

Albert Y. Kim
Assistant Professor of Statistical and Data Sciences

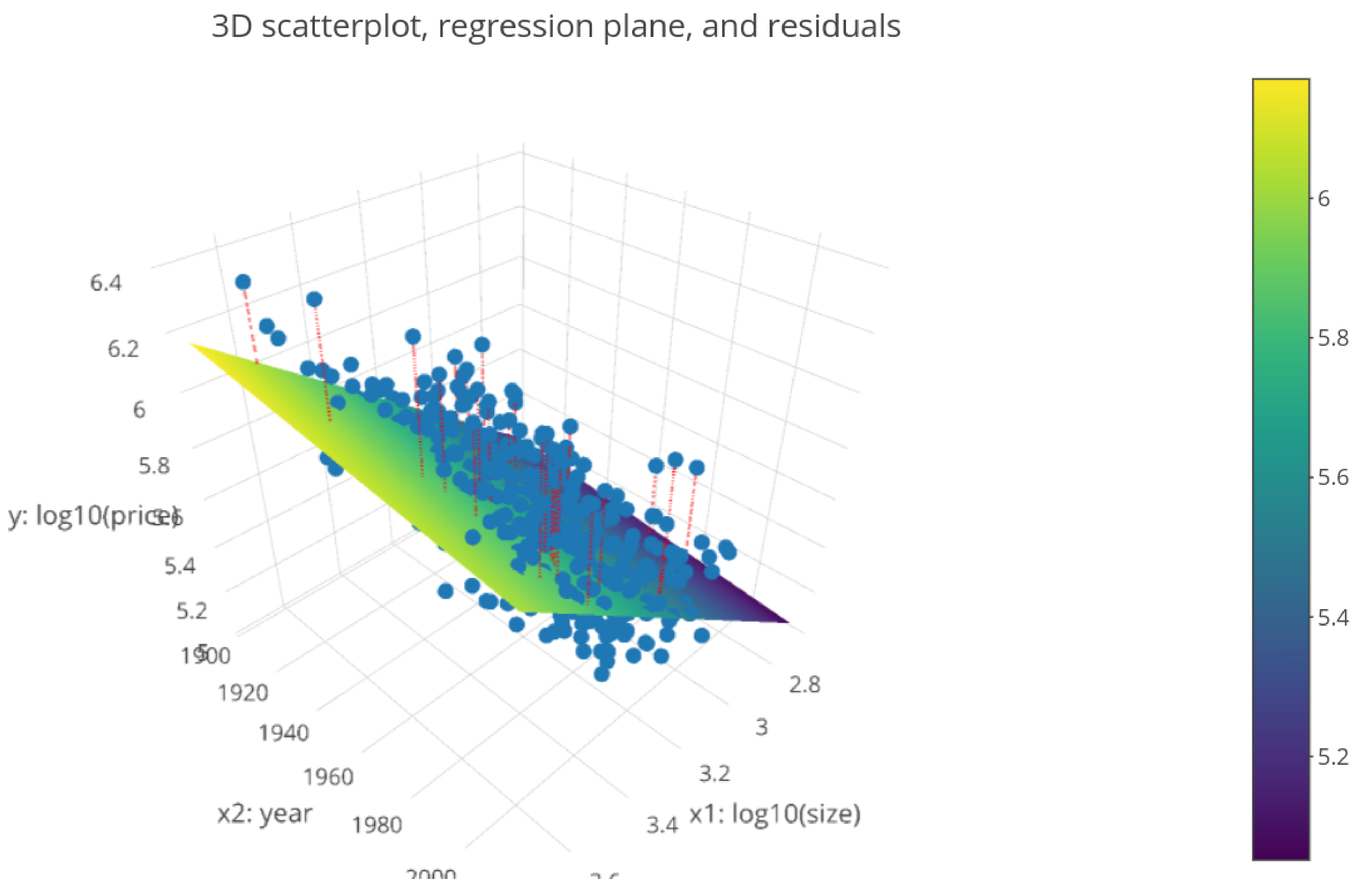
Refresher: Multiple regression

Two models with different pairs of explanatory/predictor variables:

```
# Model 1 - Two numerical:
model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                    data = house_prices)

# Model 3 - One numerical & one categorical:
model_price_3 <- lm(log10_price ~ log10_size + condition,
                    data = house_prices)
```

Refresher: Sum of squared residuals



Refresher: Sum of squared residuals

```
# Model 1
model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                    data = house_prices)
get_regression_points(model_price_1) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(sum_sq_residuals = sum(sq_residuals))
```

```
# A tibble: 1 x 1
  sum_sq_residuals
             <dbl>
1             585.
```

Refresher: Sum of squared residuals

```
# Model 3
model_price_3 <- lm(log10_price ~ log10_size + condition,
                    data = house_prices)

get_regression_points(model_price_3) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(sum_sq_residuals = sum(sq_residuals))
```

```
# A tibble: 1 x 1
  sum_sq_residuals
             <dbl>
1             608.
```

Let's practice!

Assessing model fit with R-squared

MODELING WITH DATA IN THE TIDYVERSE

Albert Y. Kim
Assistant Professor of Statistical and Data Sciences

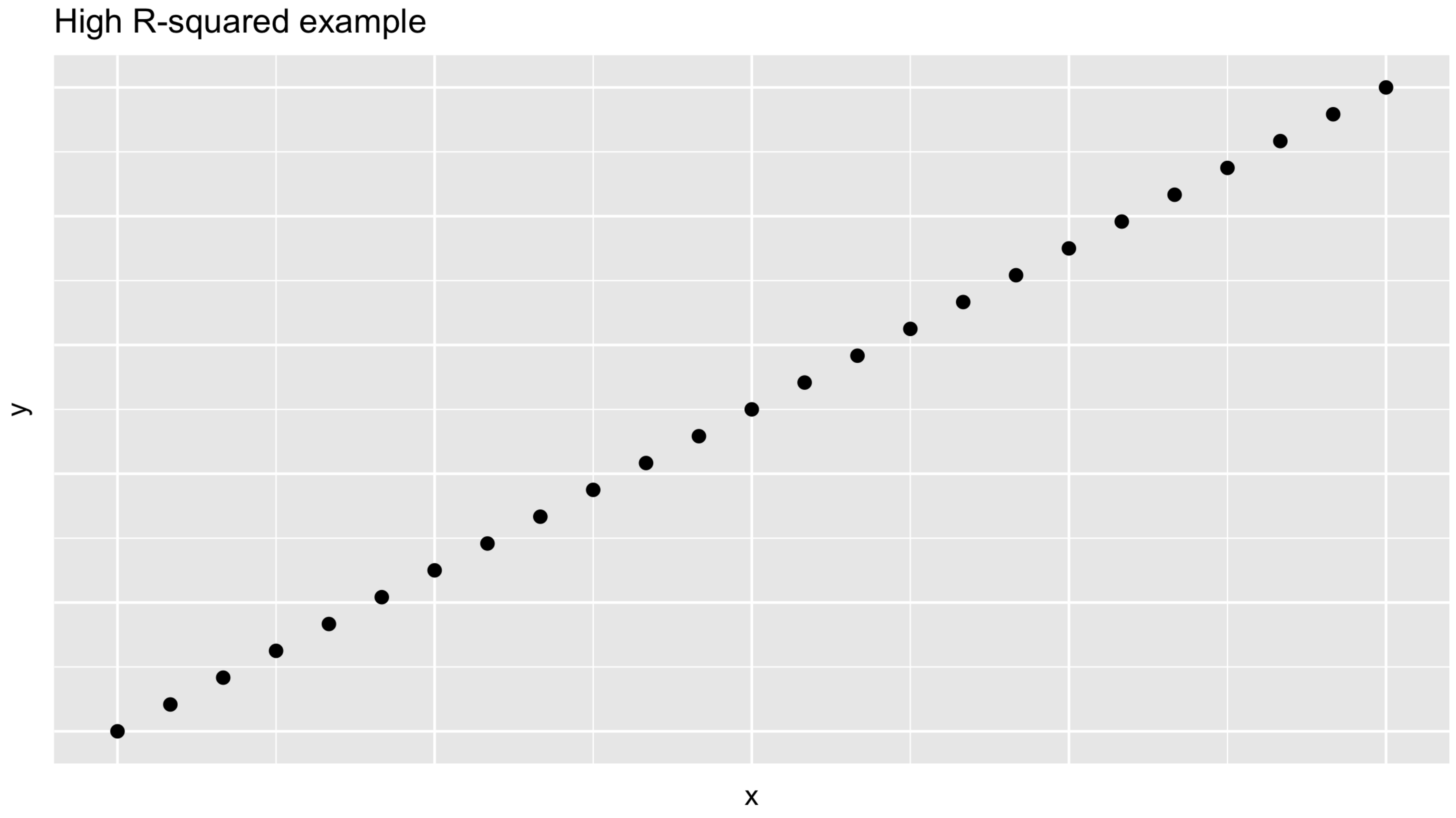
R-squared

$$R^2 = 1 - \frac{\text{Var(residuals)}}{\text{Var}(y)}$$

- R^2 is between 0 & 1
- Smaller R^2 ~ "poorer fit"
- $R^2 = 1$ ~ "perfect fit" and $R^2 = 0$ ~ "no fit"

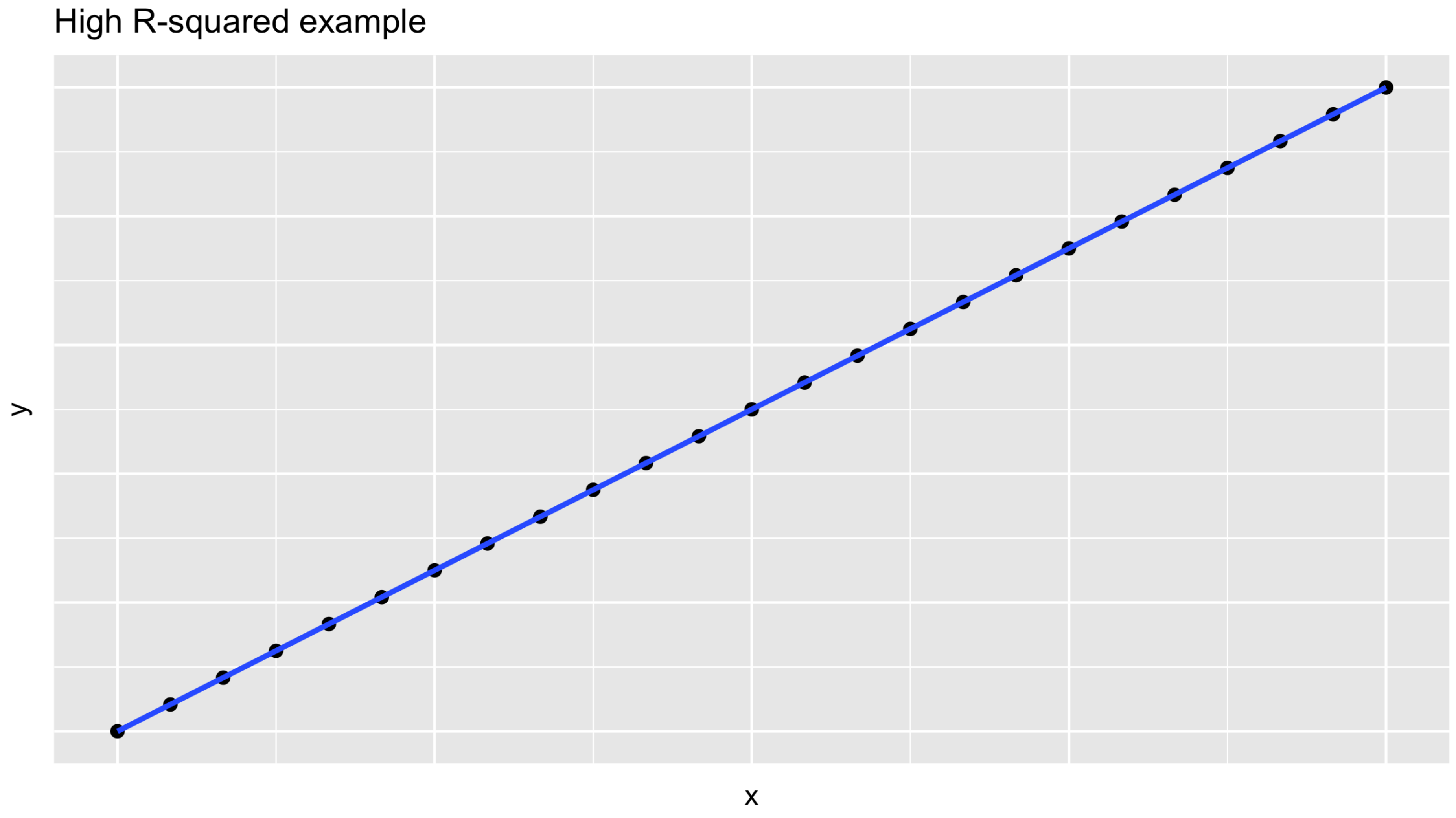
High R-squared value example

$$R^2 = 1 - \frac{\text{Var(residuals)}}{\text{Var}(y)}$$



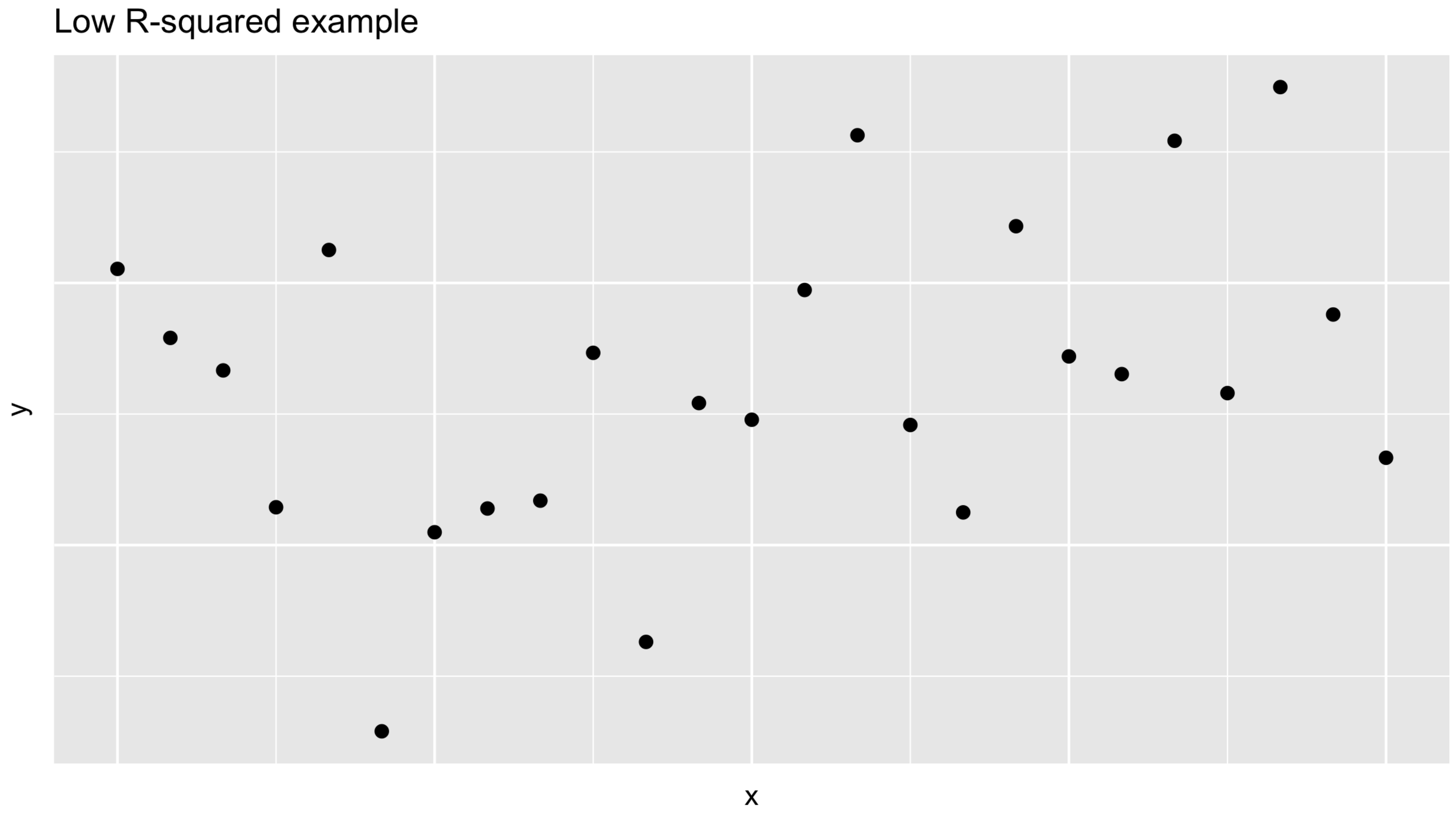
High R-squared value: "Perfect" fit

$$R^2 = 1 - \frac{\text{Var(residuals)}}{\text{Var}(y)}$$



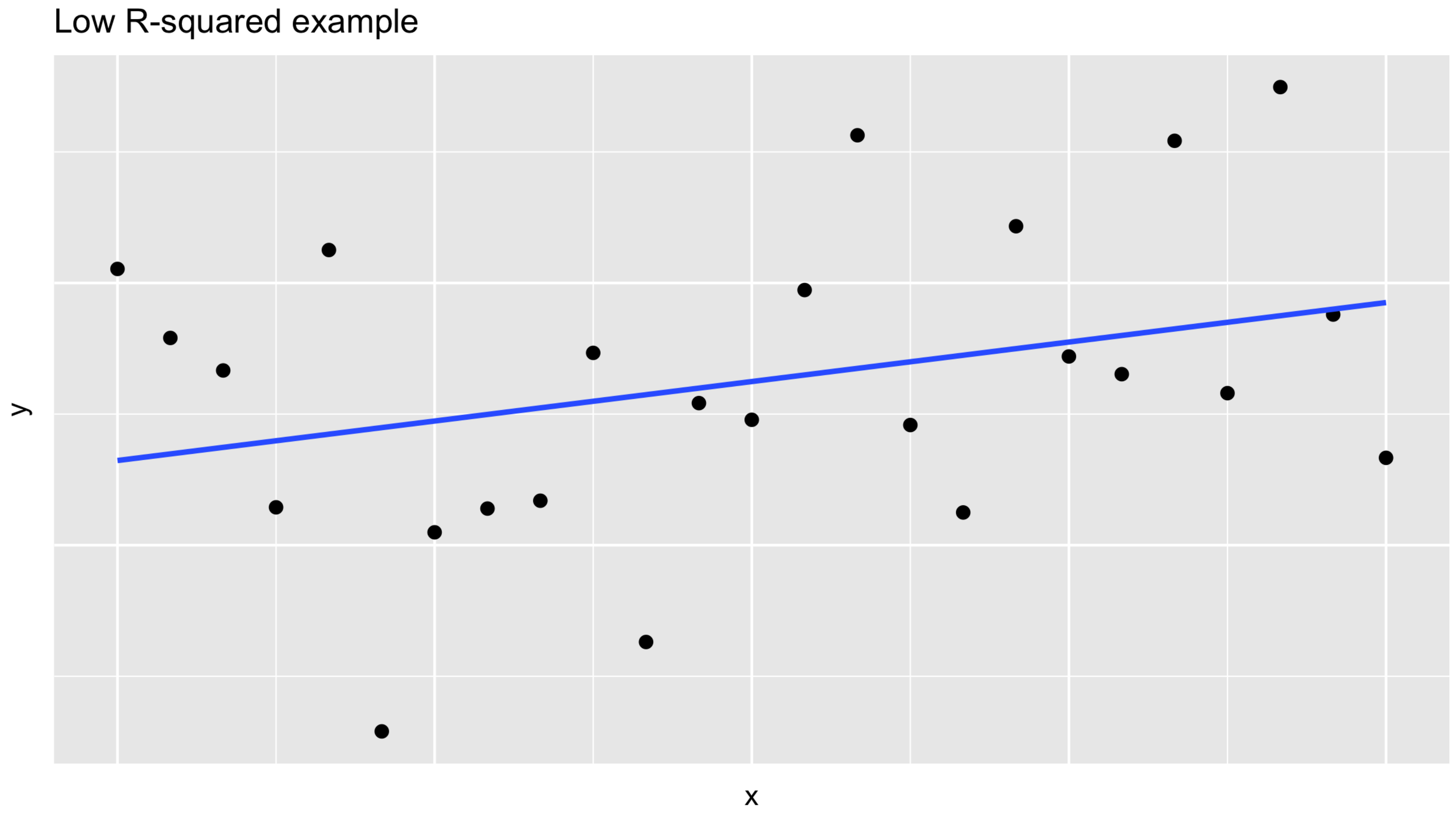
Low R-squared value example

$$R^2 = 1 - \frac{\text{Var(residuals)}}{\text{Var}(y)}$$



Low R-squared value example

$$R^2 = 1 - \frac{\text{Var(residuals)}}{\text{Var}(y)}$$



Numerical interpretation

Since $\text{Var}(y) \geq \text{Var}(\text{residuals})$ and

$$R^2 = 1 - \frac{\text{Var}(\text{residuals})}{\text{Var}(y)} = \frac{\text{Var}(y) - \text{Var}(\text{residuals})}{\text{Var}(y)}$$

R^2's interpretation is: *the proportion of the total variation in the outcome variable* y *that the model explains.*

Computing R-squared

```
# Model 1: price as a function of size and year built
model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                    data = house_prices)

get_regression_points(model_price_1) %>%
  summarize(r_squared = 1 - var(residual)/var(log10_price))
```

```
# A tibble: 1 x 1
  r_squared
      <dbl>
1     0.483
```

Computing R-squared

```
# Model 3: price as a function of size and condition
model_price_3 <- lm(log10_price ~ log10_size + condition,
                    data = house_prices)

get_regression_points(model_price_3) %>%
  summarize(r_squared = 1 - var(residual)/var(log10_price))
```

```
# A tibble: 1 x 1
  r_squared
      <dbl>
1     0.462
```

Let's practice!

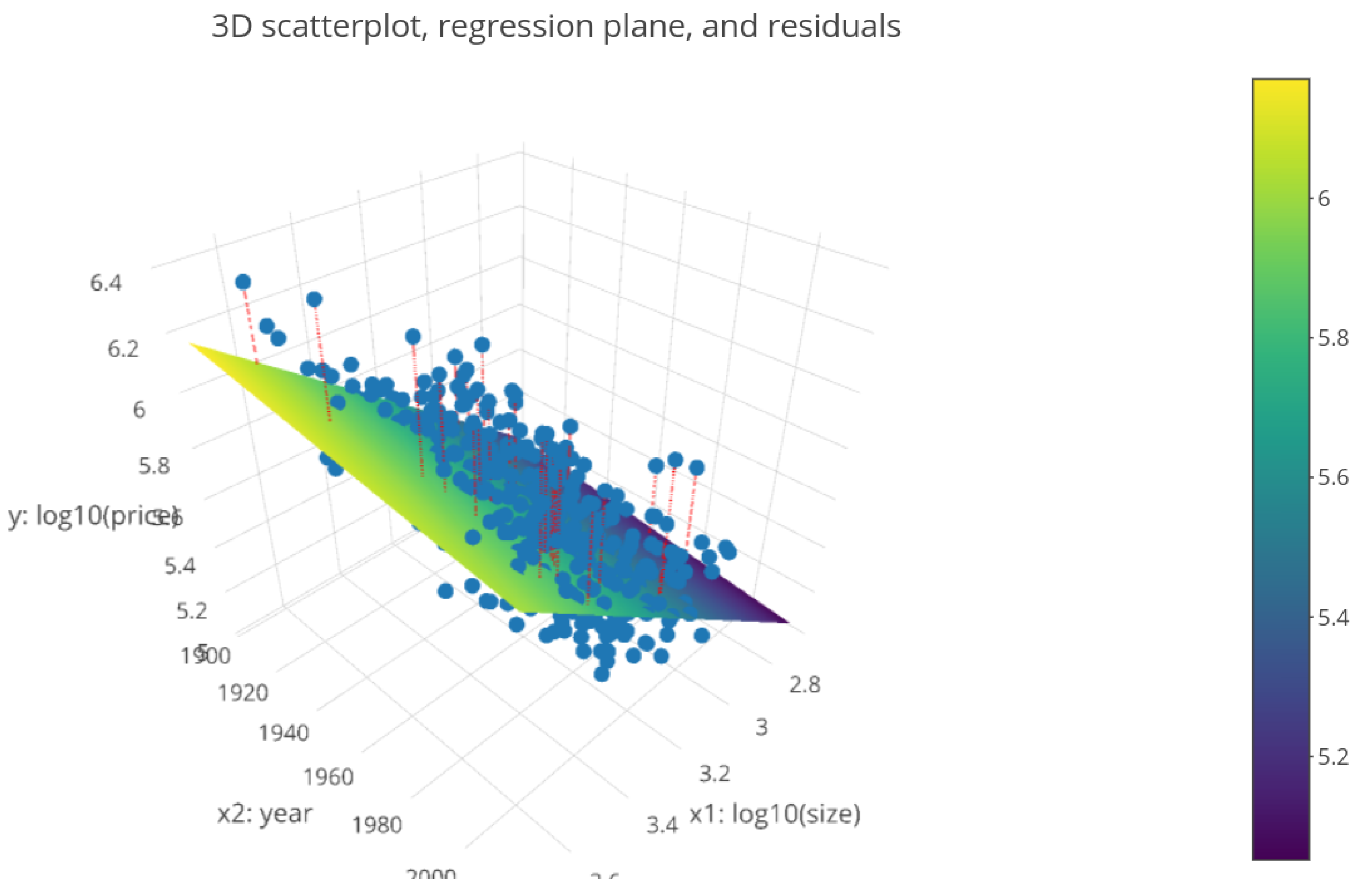
MODELING WITH DATA IN THE TIDYVERSE

Assessing predictions with RMSE

MODELING WITH DATA IN THE TIDYVERSE

Albert Y. Kim
Assistant Professor of Statistical and Data Sciences

Refresher: Residuals



Mean squared error

```
# Model 1: price as a function of size and year built
model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                    data = house_prices)

# Sum of squared residuals:
get_regression_points(model_price_1) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(sum_sq_residuals = sum(sq_residuals))
```

```
# A tibble: 1 x 1
  sum_sq_residuals
             <dbl>
1             585.
```

Mean squared error

```
# Mean squared error: use mean() instead of sum():
get_regression_points(model_price_1) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(mse = mean(sq_residuals))
```

```
# A tibble: 1 x 1
     mse
   <dbl>
1 0.0271
```

Root mean squared error

```
# Root mean squared error:
get_regression_points(model_price_1) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(mse = mean(sq_residuals)) %>%
  mutate(rmse = sqrt(mse))
```

```
# A tibble: 1 x 2
     mse  rmse
   <dbl> <dbl>
1 0.0271 0.164
```

RMSE of predictions on new houses

```
# Recreate data frame of "new" houses
new_houses <- data_frame(
  log10_size = c(2.9, 3.6),
  condition = factor(c(3, 4))
)
new_houses
```

```
# A tibble: 2 x 2
  log10_size condition
       <dbl> <fct>
1        2.9 3
2        3.6 4
```

RMSE of predictions on new houses

```
# Get predictions
get_regression_points(model_price_3,
                      newdata = new_houses)
```

```
# A tibble: 2 x 4
     ID log10_size condition log10_price_hat
  <int>      <dbl> <fct>               <dbl>
1     1        2.9 3                    5.34
2     2        3.6 4                    5.94
```

RMSE of predictions on new houses

```
# Compute RMSE
get_regression_points(model_price_3,
                      newdata = new_houses) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(mse = mean(sq_residuals)) %>%
  mutate(rmse = sqrt(mse))
```

```
Error in mutate_impl(.data, dots) :
  Evaluation error: object 'residual' not found.
```

Let's practice!

MODELING WITH DATA IN THE TIDYVERSE

Validation set prediction framework

MODELING WITH DATA IN THE TIDYVERSE

Albert Y. Kim
Assistant Professor of Statistical and Data Sciences

Validation set approach

Use two independent datasets to:

1. Train/fit your model
2. Evaluate your model's predictive power i.e. validate your model

Training/test set split

Randomly split all n observations (white) into

1. A *training set* (blue) to fit models
2. A *test set* (orange) to make predictions on

Training/test set split in R

```
library(dplyr)

# Randomly shuffle order of rows:
house_prices_shuffled <- house_prices %>%
  sample_frac(size = 1, replace = FALSE)

# Split into train and test:
train <- house_prices_shuffled %>%
  slice(1:10000)
test <- house_prices_shuffled %>%
  slice(10001:21613)
```

Training models on training data

```
train_model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                          data = train)

get_regression_table(train_model_price_1)
```

```
# A tibble: 3 x 7
  term       estimate std_error statistic p_value lower_ci...
  <chr>         <dbl>     <dbl>     <dbl>   <dbl>    <dbl>...
1 intercept     5.34      0.111      48.3       0    5.13...
2 log10_size    0.923     0.009      97.5       0    0.905...
3 yr_built     -0.001     0         -23.0       0   -0.001...
```

Making predictions on test data

```
# Train model on train:
train_model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                          data = train)

# Get predictions on test:
get_regression_points(train_model_price_1, newdata = test)
```

```
# A tibble: 11,613 x 6
      ID log10_price log10_size yr_built log10_price_hat...
   <int>       <dbl>      <dbl>    <dbl>            <dbl>...
 1     1        5.83       3.29     1951             5.71...
 2     2        5.88       3.40     1922             5.84...
 3     3        6.15       3.67     2002             5.99...
 4     4        5.62       3        1953             5.43...
...
# ... with 11,603 more rows
```

Assessing predictions with RMSE

```
# Train model:
train_model_price_1 <- lm(log10_price ~ log10_size + yr_built,
                          data = train)

# Get predictions and compute RMSE:
get_regression_points(train_model_price_1, newdata = test) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(rmse = sqrt(mean(sq_residuals)))
```

```
# A tibble: 1 x 1
   rmse
  <dbl>
1 0.165
```

Comparing RMSE

```
# Train model:
train_model_price_3 <- lm(log10_price ~ log10_size + condition,
                          data = train)

# Get predictions and compute RMSE:
get_regression_points(train_model_price_3, newdata = test) %>%
  mutate(sq_residuals = residual^2) %>%
  summarize(rmse = sqrt(mean(sq_residuals)))
```

```
# A tibble: 1 x 1
   rmse
  <dbl>
1 0.168
```

Let's practice!

MODELING WITH DATA IN THE TIDYVERSE

Conclusion - Where to go from here?

MODELING WITH DATA IN THE TIDYVERSE

Albert Y. Kim
Assistant Professor of Statistical and Data Sciences

R source code for all videos

Available at http://bit.ly/modeling_tidyverse

R source code for "Modeling with Data in the Tidyverse" DataCamp course

modeling_with_data_tidyverse.R

```
# R source code for all slides/videos in Albert Y. Kim's "Modeling with Data in
# the Tidyverse" DataCamp course:

# Load all necessary packages -----
library(ggplot2)
library(dplyr)
library(moderndive)

# Chapter 1 - Video 1: Background on modeling for explanation -----
## Modeling for explanation example
glimpse(evals)

## Exploratory data analysis
ggplot(evals, aes(x = score)) +
  geom_histogram(binwidth = 0.25) +
  labs(x = "teaching score", y = "count")

```

Other Tidyverse courses

Available **here** and **here**

SKILL TRACK

Tidyverse Fundamentals with R

Experience the whole data science pipeline from importing and tidying data to wrangling and visualizing data to modeling and communicating with data. Gain exposure to each component of this pipeline from a variety of different perspectives in this tidyverse R track.

SKILL TRACK

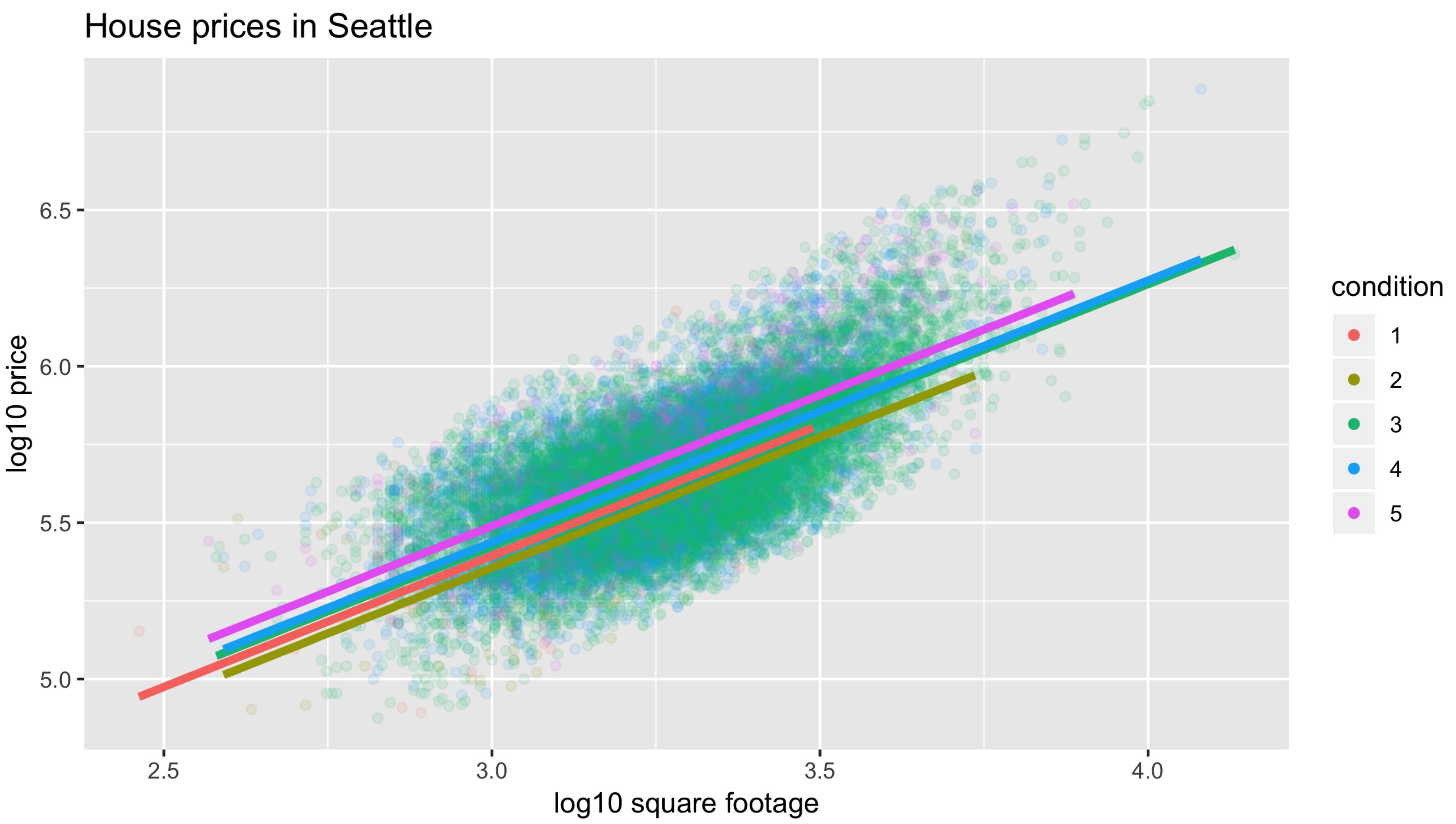
Intermediate Tidyverse Toolbox

Take your tidyverse skills to the next level. This track covers getting your data in the right condition to start your analyses, writing better code with functional programming, and generating, exploring, and evaluating machine learning models. And you'll do all of this in the wonderful and clean world of the tidyverse.

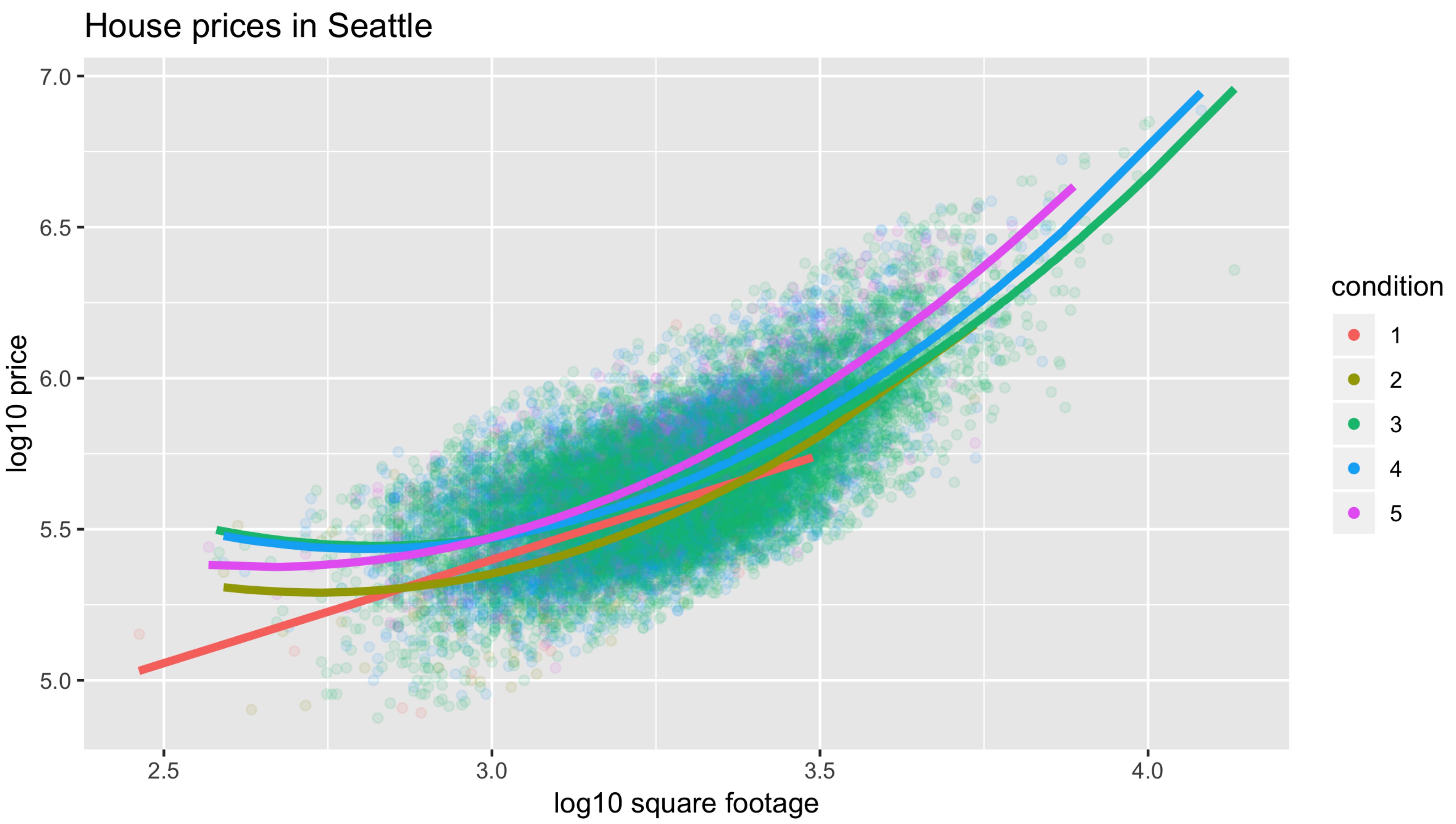
Refresher: General modeling framework

- In general: $y = f(\vec{x}) + \epsilon$
- Linear regression models: $y = \beta_0 + \beta_1 \cdot x_1 + \epsilon$

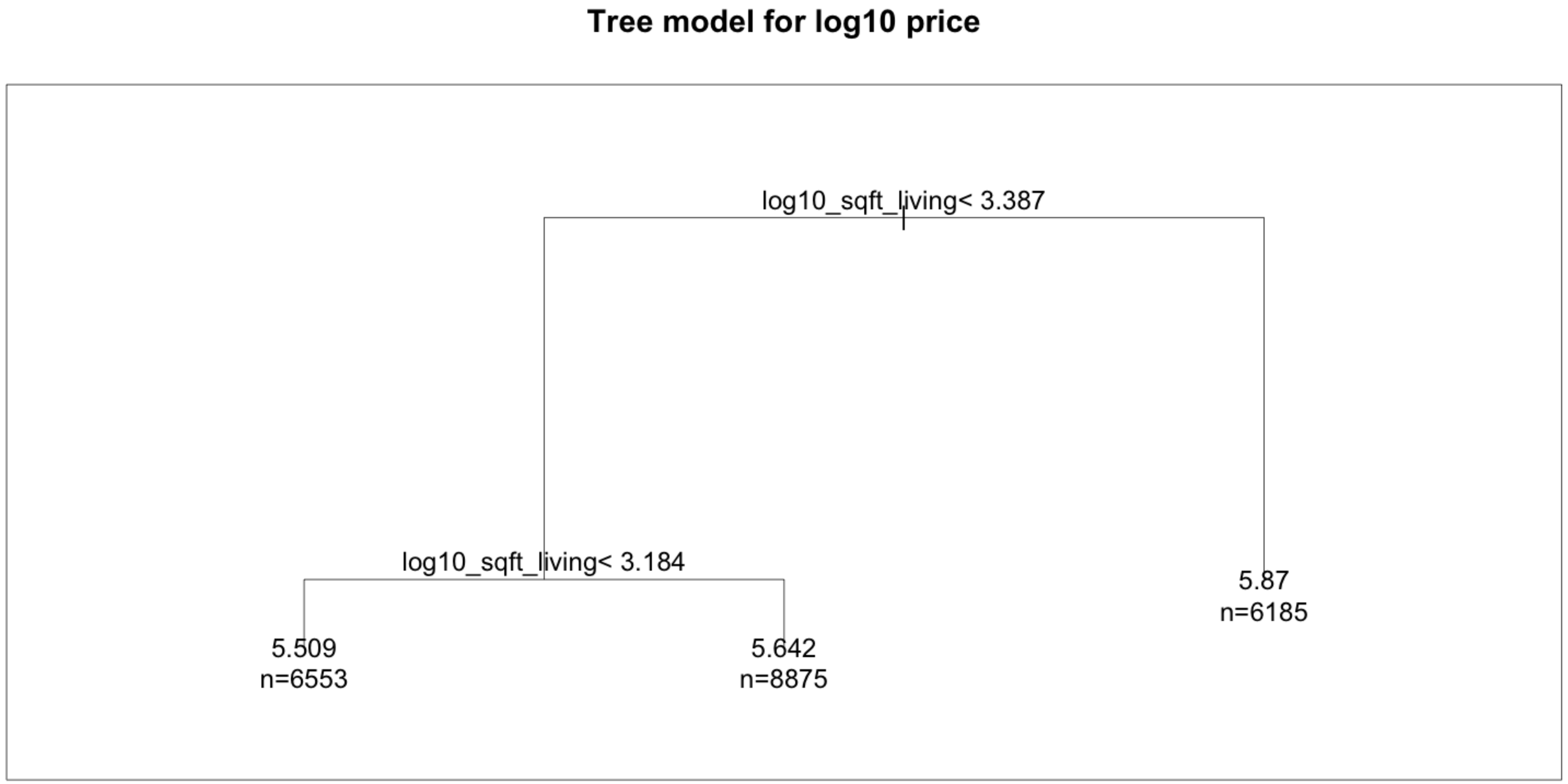
Parallel slopes model



Polynomial model



Tree models


Tree model for log10 price
log10_sqft_living< 3.387
log10_sqft_living< 3.184
5.509
n=6553
5.642
n=8875
5.87
n=6185

DataCamp courses using other models

Courses with different $f()$ in $y = f(\vec{x}) + \epsilon$:

- Machine Learning with Tree-Based Models in R
- Supervised Learning in R: Case Studies

Refresher: Regression table

```
# Fit model:
model_score_1 <- lm(score ~ age, data = evals)

# Output regression table:
get_regression_table(model_score_1)
```

```
# A tibble: 2 x 7
  term      estimate std_error statistic p_value lower_ci upper_ci
  <chr>        <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
1 intercept    4.46      0.127     35.2    0         4.21     4.71
2 age         -0.006     0.003     -2.31   0.021    -0.011   -0.001
```

ModernDive: Online textbook

ModernDive

- Uses `tidyverse` tools: `ggplot2` and `dplyr`
- Expands on the regression models from this course
- Uses `evals` and `house_prices` datasets (and more)
- **Goal**: Statistical inference via data science
- Available at **ModernDive.com**

Good luck!

MODELING WITH DATA IN THE TIDYVERSE