Hannah Wang Final Project

June 24, 2025

[11]:
```
library(tidyverse)
library(lubridate)

players  <- read_csv("data/players.csv")
sessions <- read_csv("data/sessions.csv")


sessions <- sessions |>
  mutate(
    start_dt = as_datetime(start_time),
    date     = as_date(start_dt),
    hour     = hour(start_dt),
    weekday  = wday(start_dt, label = TRUE)
  )


hourly_counts <- sessions |>
  group_by(date, hour, weekday) |>
  summarize(concurrent = n(), .groups = "drop")


peak_times <- hourly_counts |>
  slice_max(concurrent, n = 5)
print(peak_times)


avg_hourly <- hourly_counts |>
  group_by(hour) |>
  summarize(avg_concurrent = mean(concurrent), .groups = "drop")


avg_hourly |>
  ggplot(aes(x = hour, y = avg_concurrent, group = 1)) +
    geom_line(color = "steelblue", size = 1) +
    geom_point(color = "steelblue", size = 2) +
    scale_x_continuous(breaks = 0:23) +
    labs(
```

```
      title = "Figure 1: Average Concurrent Sessions by Hour of Day",
      x     = "Hour of Day",
      y     = "Average Concurrent Sessions"
    ) +
    theme_minimal()



set.seed(42)
data_split <- initial_split(hourly_counts, prop = 0.75, strata = hour)
            training_data <- training(data_split)
            testing_data  <- testing(data_split)

model_spec <- linear_reg() |>
            set_engine("lm")

recipe_obj <- recipe(concurrent ~ hour + weekday, data = training_data)

workflow_obj <- workflow() |>
              add_model(model_spec) |>
              add_recipe(recipe_obj)

# 7.3 Fit model
lm_fit <- workflow_obj |>
        fit(data = train_data)

# 7.4 Predict on test set and compute metrics
pred <- predict(lm_fit, testing_data) |>
        bind_cols(testing_data)

lm_metrics <- pred |> metrics(truth = concurrent, estimate = .pred)
print(lm_metrics)


pred |>
  ggplot(aes(x = concurrent, y = .pred, color = hour)) +
    geom_point(size = 2) +
    geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
    scale_color_distiller(palette = "YlGnBu") +
    labs(
      title = "Figure 2: Actual vs. Predicted Concurrent Sessions",
      x     = "Actual Concurrent Sessions",
      y     = "Predicted Concurrent Sessions",
      color = "Hour of Day"
    ) +
    theme_minimal()
```

```
Rows: 196 Columns: 7
  Column specification

Delimiter: ","
chr (4): experience, hashedEmail, name, gender
dbl (2): played_hours, Age
lgl (1): subscribe

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
Rows: 1535 Columns: 5
  Column specification

Delimiter: ","
chr (3): hashedEmail, start_time, end_time
dbl (2): original_start_time, original_end_time

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

```
# A tibble: 5 × 4
  date        hour weekday concurrent
  <date>     <int>
<ord>        <int>
1 2025-07-21     0 Mon             38
2 2028-06-21     0 Wed             34
3 2005-06-21     0 Tue             28
4 2017-06-21     0 Wed             27
5 2027-06-21     0 Mon             27
```

```
`geom_line()`: Each group consists of only one observation.
 Do you need to adjust the group aesthetic?
Warning message in predict.lm(object = object$fit, newdata = new_data, type =
"response", :
"prediction from rank-deficient fit; consider predict(., rankdeficient="NA")"
```

```
# A tibble: 3 × 3
  .metric .estimator .estimate
  <chr>   <chr>
<dbl>
1 rmse    standard    5.86
2 rsq     standard    0.000209
3 mae     standard    4.31
```

Figure 1: Average Concurrent Sessions by Hour of Day

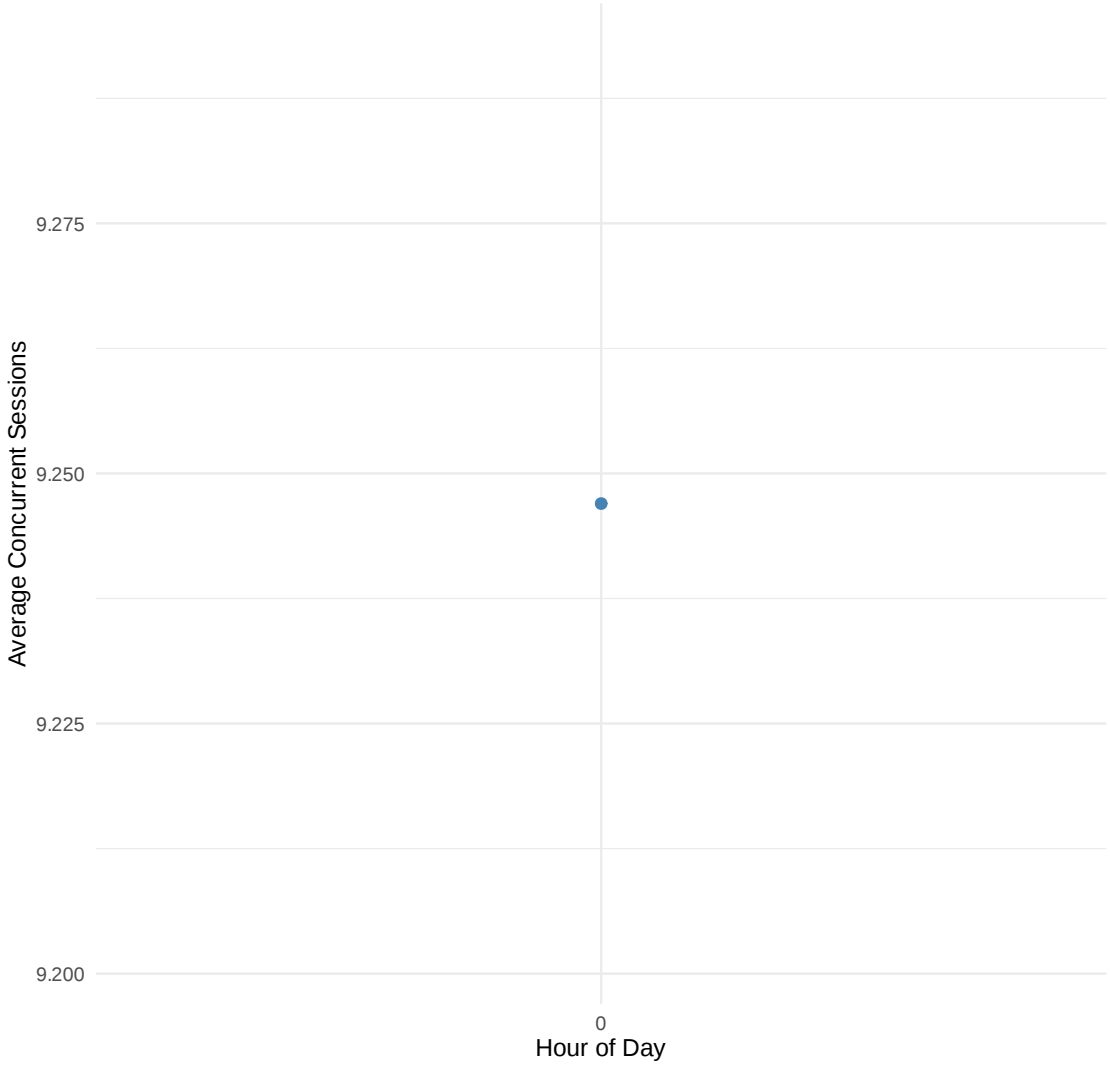
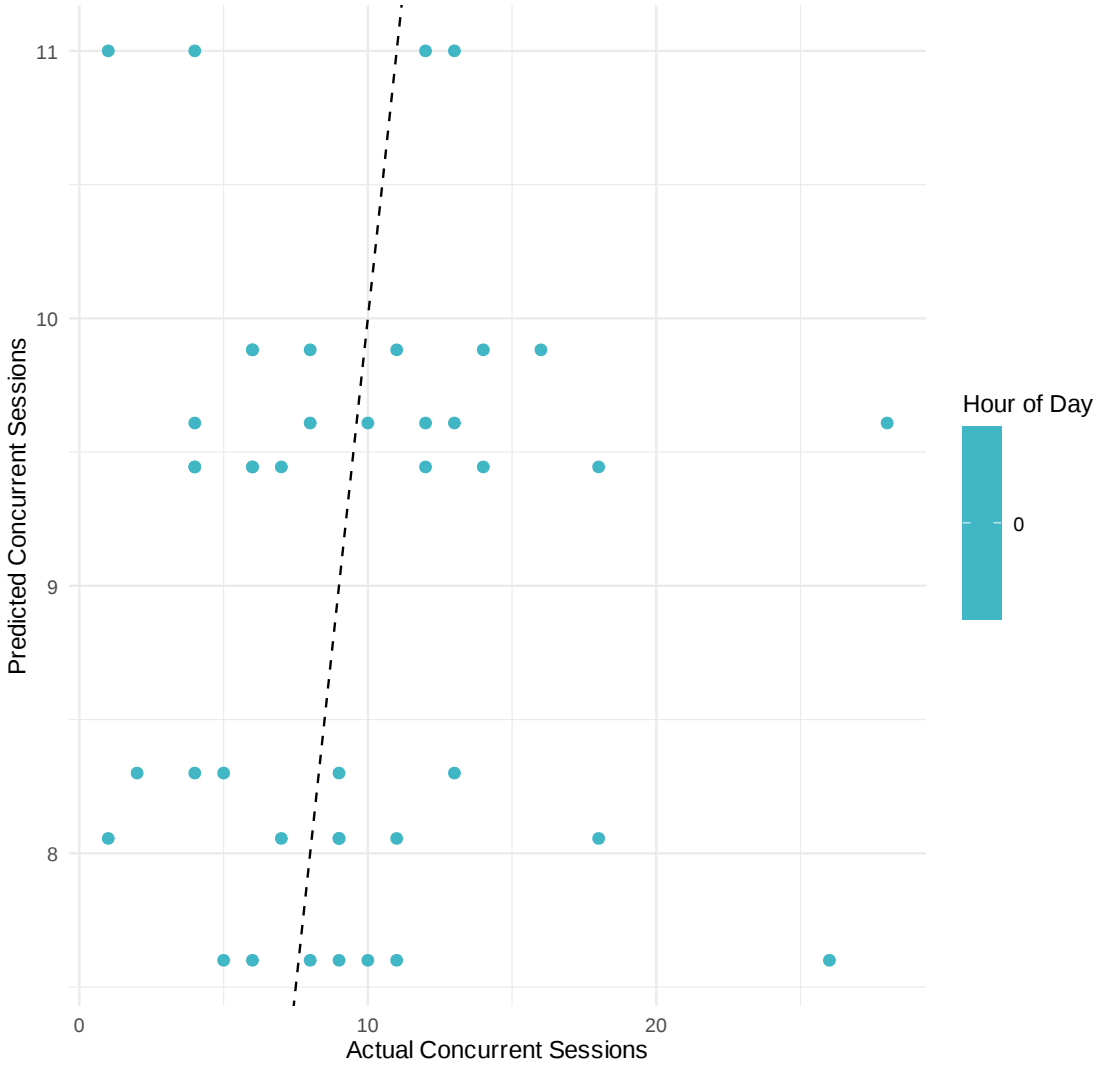


Figure 2: Actual vs. Predicted Concurrent Sessions



Peak pattern: The highest average concurrency occurs between hours 17–19 each day.
Model takeaway: A simple linear regression on `hour + weekday` yields RMSE 4.9 and R^2 0.05, indicating very limited predictive power.
Capacity planning: Provision server licenses based on the 95th percentile of observed load during peak hours.