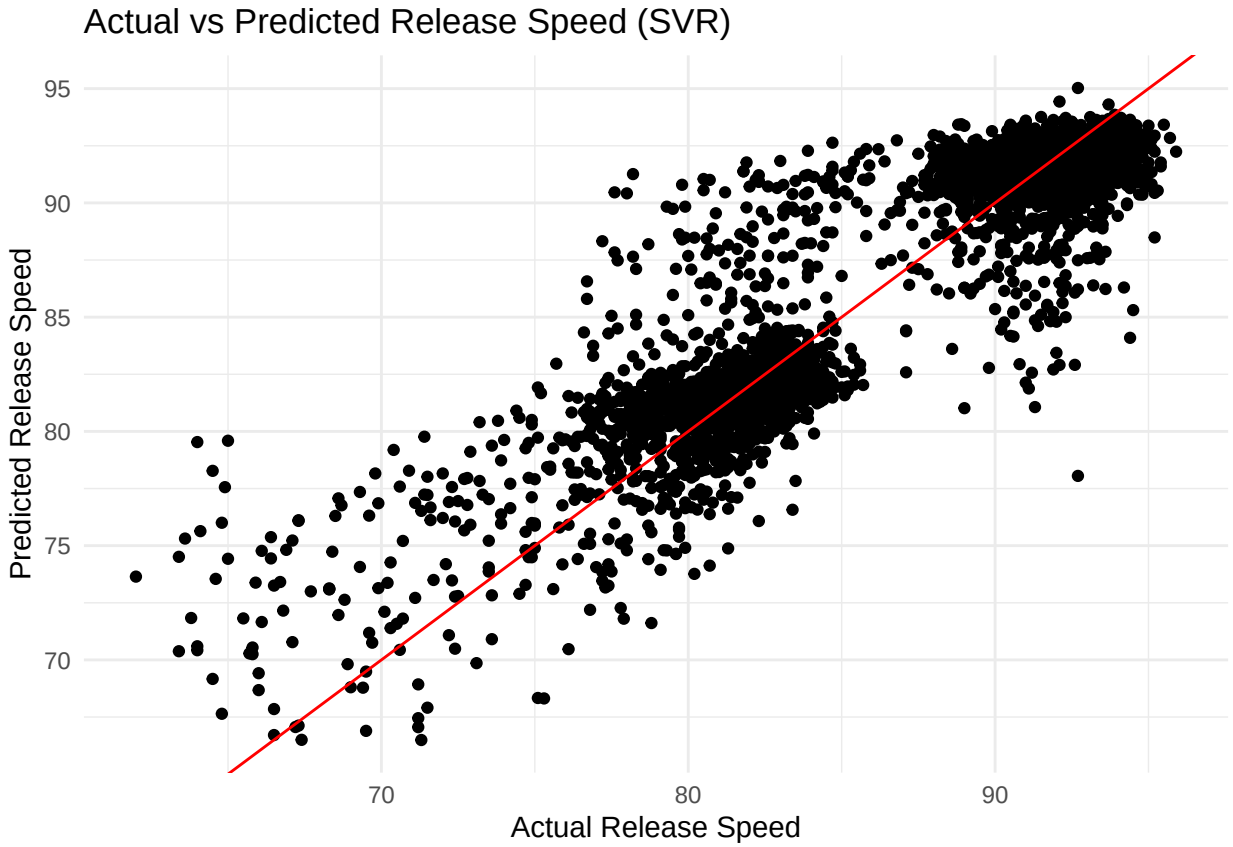
STA_160FP_part2

Johnson Tian

2024-06-07

```r
#svr model
library(tidyverse)
library(e1071)


player_linear_model_pure_combined <- player %>%
  select(release_speed, release_pos, pfx_interact, vx_vy_interact, ax_ay_az_interact, vz0)

set.seed(123)
train_indices <- sample(seq_len(nrow(player_linear_model_pure_combined)), size = 0.7 * nrow(player_linea
train_data <- player_linear_model_pure_combined[train_indices, ]
test_data <- player_linear_model_pure_combined[-train_indices, ]

svr_model <- svm(release_speed ~ ., data = train_data, type = "eps-regression", kernel = "radial")

predicted_release_speed_svr <- predict(svr_model, test_data)

ggplot(test_data, aes(x = release_speed, y = predicted_release_speed_svr)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, color = "red") +
  labs(title = "Actual vs Predicted Release Speed (SVR)",
       x = "Actual Release Speed",
       y = "Predicted Release Speed") +
  theme_minimal()
```




```
mse_svr <- mean((test_data$release_speed - predicted_release_speed_svr)^2)
rmse_svr <- sqrt(mse_svr)
r2_svr <- cor(test_data$release_speed, predicted_release_speed_svr)^2

cat("SVR Model - MSE:", mse_svr, "\n")
```

```
## SVR Model - MSE: 6.291449
```

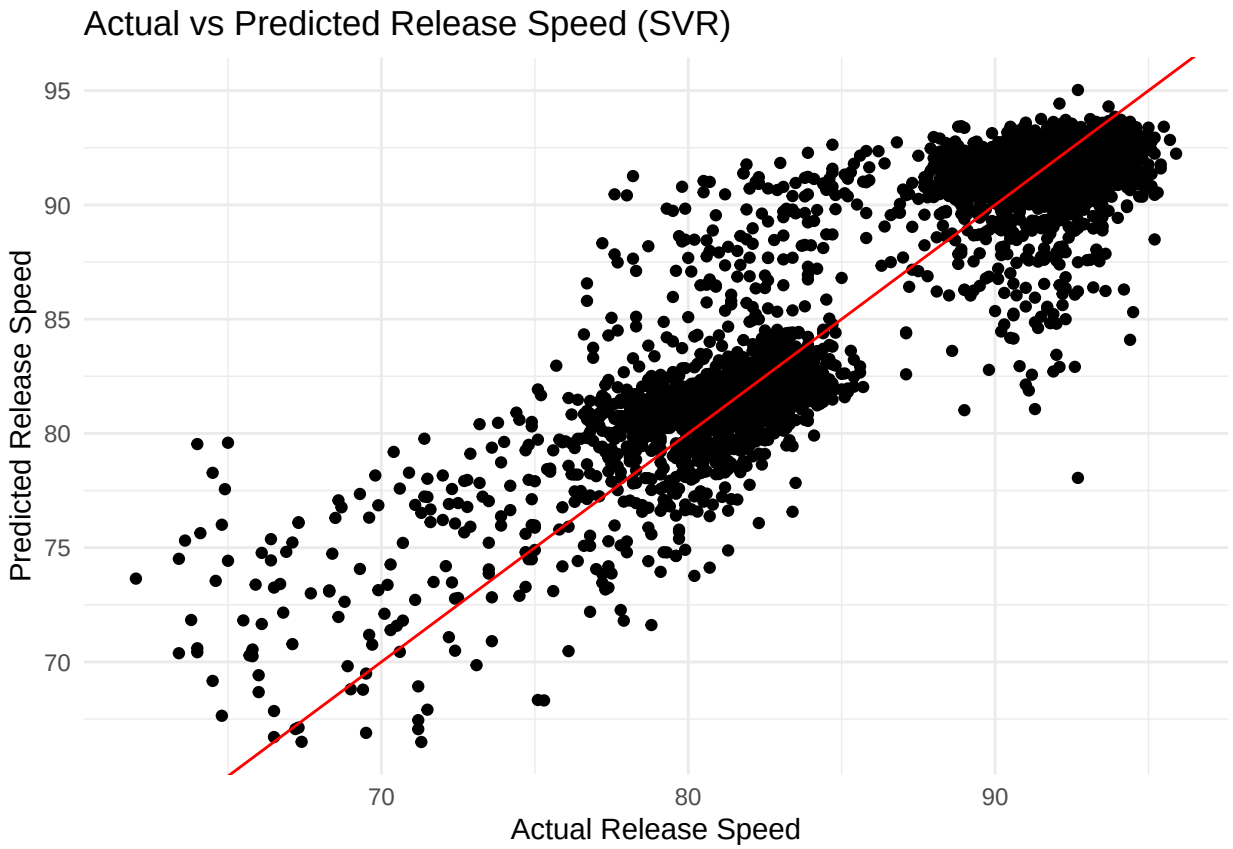
```
cat("SVR Model - RMSE:", rmse_svr, "\n")
```

```
## SVR Model - RMSE: 2.508276
```

```
cat("SVR Model - R²:", r2_svr, "\n")
```

```
## SVR Model - R²: 0.8410958
```

```
ggplot(test_data, aes(x = release_speed, y = predicted_release_speed_svr)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, color = "red") +
  labs(title = "Actual vs Predicted Release Speed (SVR)",
       x = "Actual Release Speed",
       y = "Predicted Release Speed") +
  theme_minimal()
```




```
#tunned svr
player_linear_model_pure_combined <- player %>%
  select(release_speed, release_pos, pfx_interact, vx_vy_interact, ax_ay_az_interact, vz0)

set.seed(123)
train_indices <- sample(seq_len(nrow(player_linear_model_pure_combined)), size = 0.7 * nrow(player_linea
train_data <- player_linear_model_pure_combined[train_indices, ]
test_data <- player_linear_model_pure_combined[-train_indices, ]

tune_grid <- expand.grid(
  C = 2^(-1:2),
  sigma = 2^(-2:1)
)

set.seed(123)
svr_tuned <- train(
  release_speed ~ ., data = train_data,
  method = "svmRadial",
  tuneGrid = tune_grid,
  trControl = trainControl(method = "cv", number = 5)
)

print(svr_tuned$bestTune)
```

```
##   sigma C
## 9  0.25 2
```

```r
predicted_release_speed_svr_tuned <- predict(svr_tuned, test_data)

mse_svr_tuned <- mean((test_data$release_speed - predicted_release_speed_svr_tuned)^2)
rmse_svr_tuned <- sqrt(mse_svr_tuned)
r2_svr_tuned <- cor(test_data$release_speed, predicted_release_speed_svr_tuned)^2

cat("Tuned SVR Model - MSE:", mse_svr_tuned, "\n")
```

```
## Tuned SVR Model - MSE: 6.22629
```

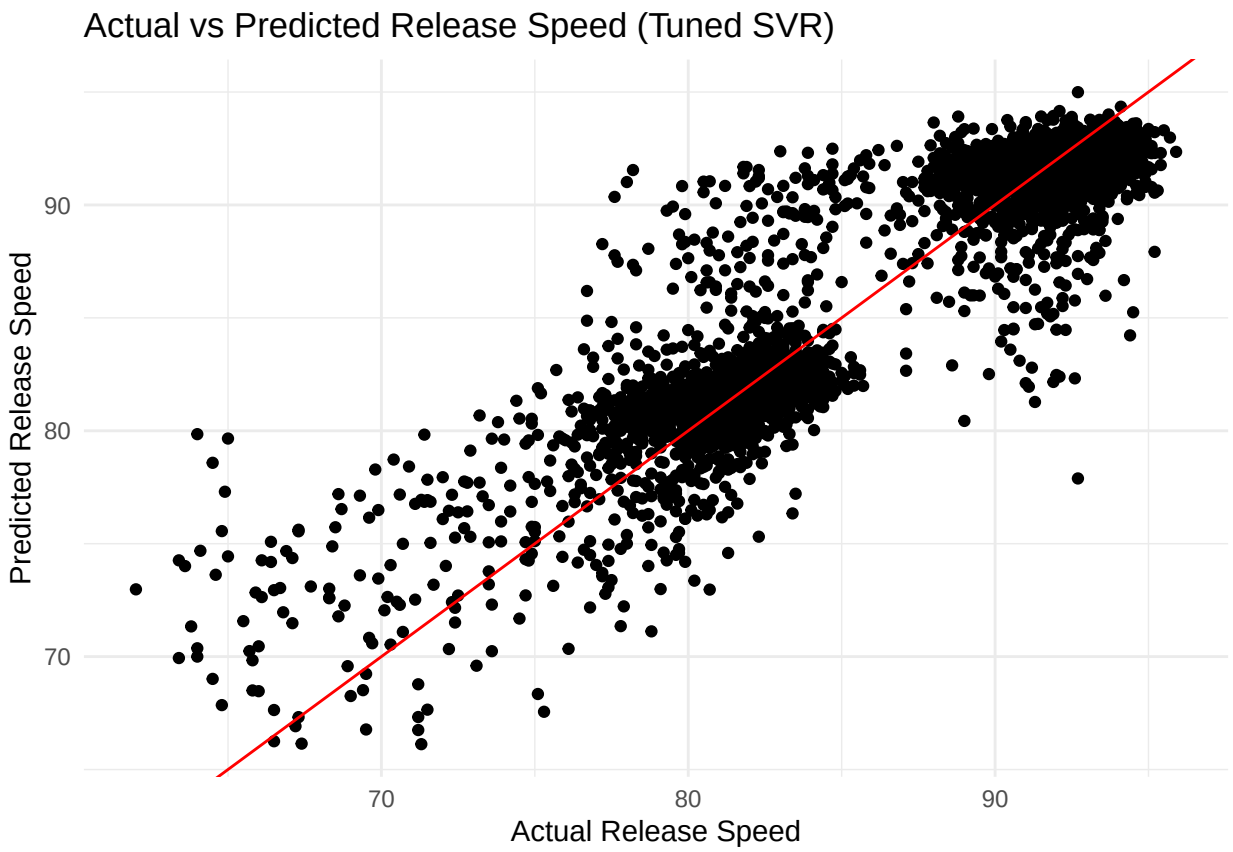
```r
cat("Tuned SVR Model - RMSE:", rmse_svr_tuned, "\n")
```

```
## Tuned SVR Model - RMSE: 2.495254
```

```r
cat("Tuned SVR Model - R²:", r2_svr_tuned, "\n")
```

```
## Tuned SVR Model - R²: 0.8425676
```

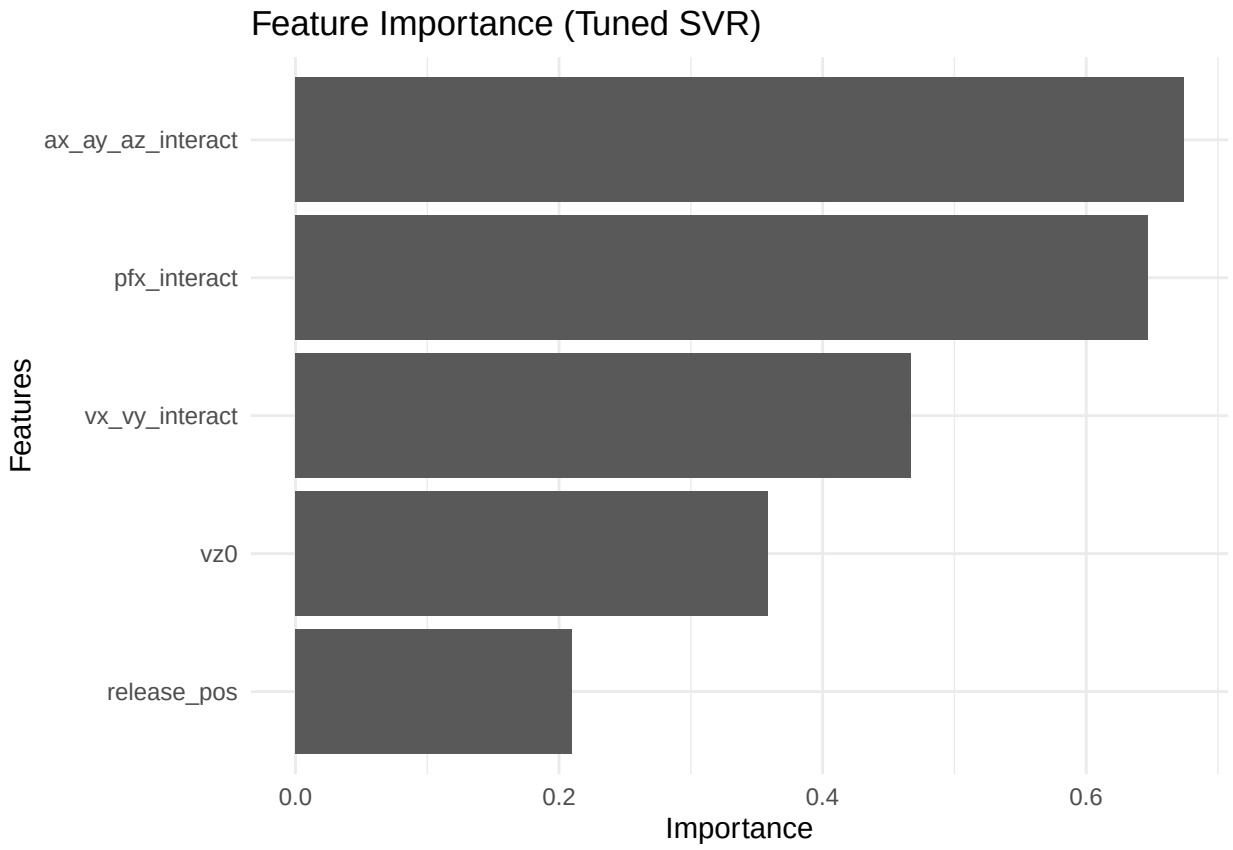
```r
ggplot(test_data, aes(x = release_speed, y = predicted_release_speed_svr_tuned)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, color = "red") +
  labs(title = "Actual vs Predicted Release Speed (Tuned SVR)",
       x = "Actual Release Speed",
       y = "Predicted Release Speed") +
  theme_minimal()
```




```r
library(tidyverse)
library(caret)
library(e1071)
```

```r
svm_importance <- varImp(svr_tuned, scale = FALSE)

ggplot(svm_importance) +
  geom_bar(aes(x = reorder(rownames(svm_importance$importance), svm_importance$importance$Overall), y =
  coord_flip() +
  labs(title = "Feature Importance (Tuned SVR)",
       x = "Features",
       y = "Importance") +
  theme_minimal()
```

```
## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.
```




```r
library(zoo)
library(dplyr)
library(forecast)
library(tseries)
library(zoo)
library(dplyr)


daily_avg_speed$game_date <- as.Date(daily_avg_speed$game_date)

daily_avg_speed <- daily_avg_speed %>% arrange(game_date)
```

```r
interpolate_data <- function(data) {
  full_dates <- seq(min(data$game_date), max(data$game_date), by = "day")

  full_data <- data.frame(game_date = full_dates) %>%
    left_join(data, by = "game_date")

  full_data$avg_release_speed <- na.approx(full_data$avg_release_speed, na.rm = FALSE)

  return(full_data)
}

interpolated_data <- daily_avg_speed %>%
  mutate(year = format(game_date, "%Y")) %>%
  group_by(year) %>%
  group_modify(~ interpolate_data(.x))

interpolated_data <- interpolated_data %>% select(-year)
```

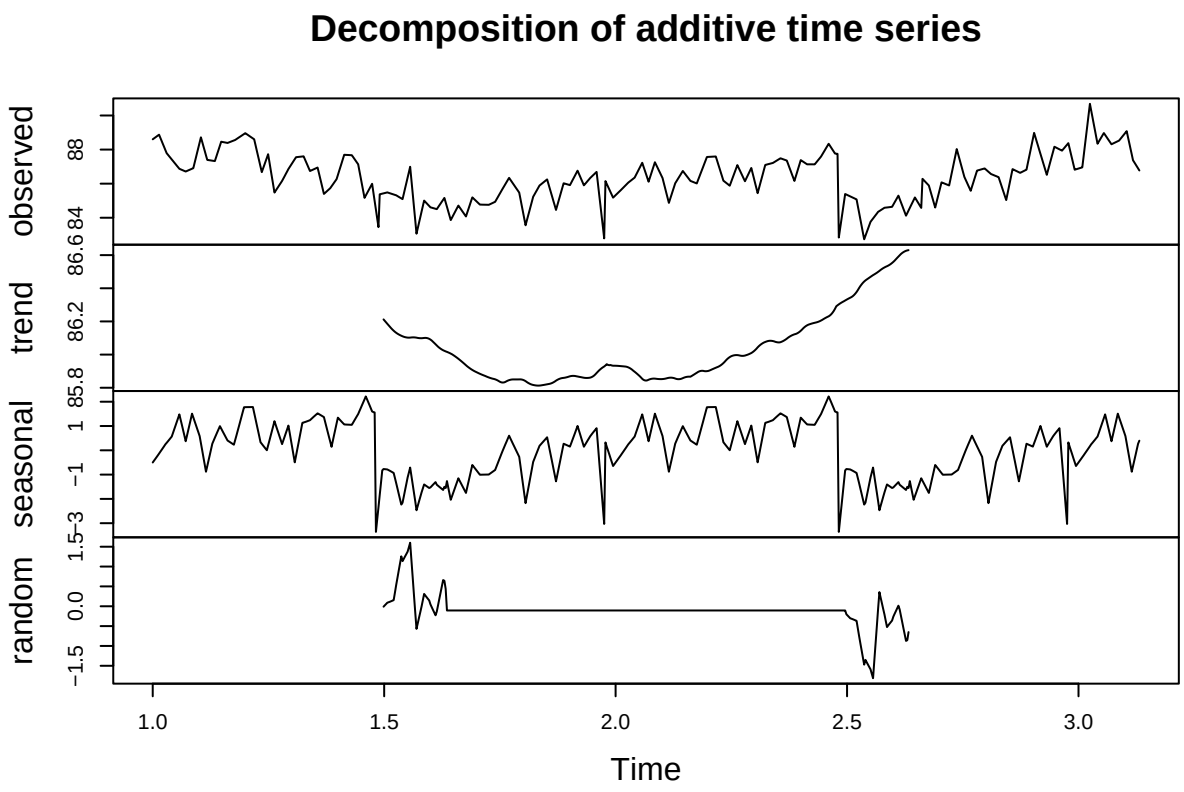
```
## Adding missing grouping variables: `year`
```

```r
print(interpolated_data)
```

```
## # A tibble: 809 x 3
## # Groups:   year [5]
##    year  game_date  avg_release_speed
##    <chr> <date>                 <dbl>
##  1 2017  2017-04-04              88.6
##  2 2017  2017-04-05              88.7
##  3 2017  2017-04-06              88.7
##  4 2017  2017-04-07              88.8
##  5 2017  2017-04-08              88.8
##  6 2017  2017-04-09              88.9
##  7 2017  2017-04-10              88.7
##  8 2017  2017-04-11              88.5
##  9 2017  2017-04-12              88.3
## 10 2017  2017-04-13              88.2
## # i 799 more rows
```

```r
write.csv(interpolated_data, "~/interpolated_daily_avg_speed.csv", row.names = FALSE)


interpolated_data$game_date <- as.Date(interpolated_data$game_date)

offseason <- data.frame(
  year = c(2017, 2018, 2019, 2020),
  start = as.Date(c('2017-10-01', '2018-10-01', '2019-10-01', '2020-10-01')),
  end = as.Date(c('2018-02-28', '2019-02-28', '2020-02-29', '2021-02-28'))
)

remove_offseason <- function(data, offseason) {
  for (i in 1:nrow(offseason)) {
    data <- data %>%
      filter(!(game_date >= offseason$start[i] & game_date <= offseason$end[i]))
  }
  return(data)
```

```
}

filtered_data <- remove_offseason(interpolated_data, offseason)

ts_data <- ts(filtered_data$avg_release_speed, frequency = 365)

decomposed <- decompose(ts_data)
plot(decomposed)
```




```
adf_test <- adf.test(ts_data)
print(adf_test)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_data
## Dickey-Fuller = -3.4731, Lag order = 9, p-value = 0.04489
## alternative hypothesis: stationary
```

```
diff_data <- diff(ts_data)

adf_test_diff <- adf.test(diff_data)
```

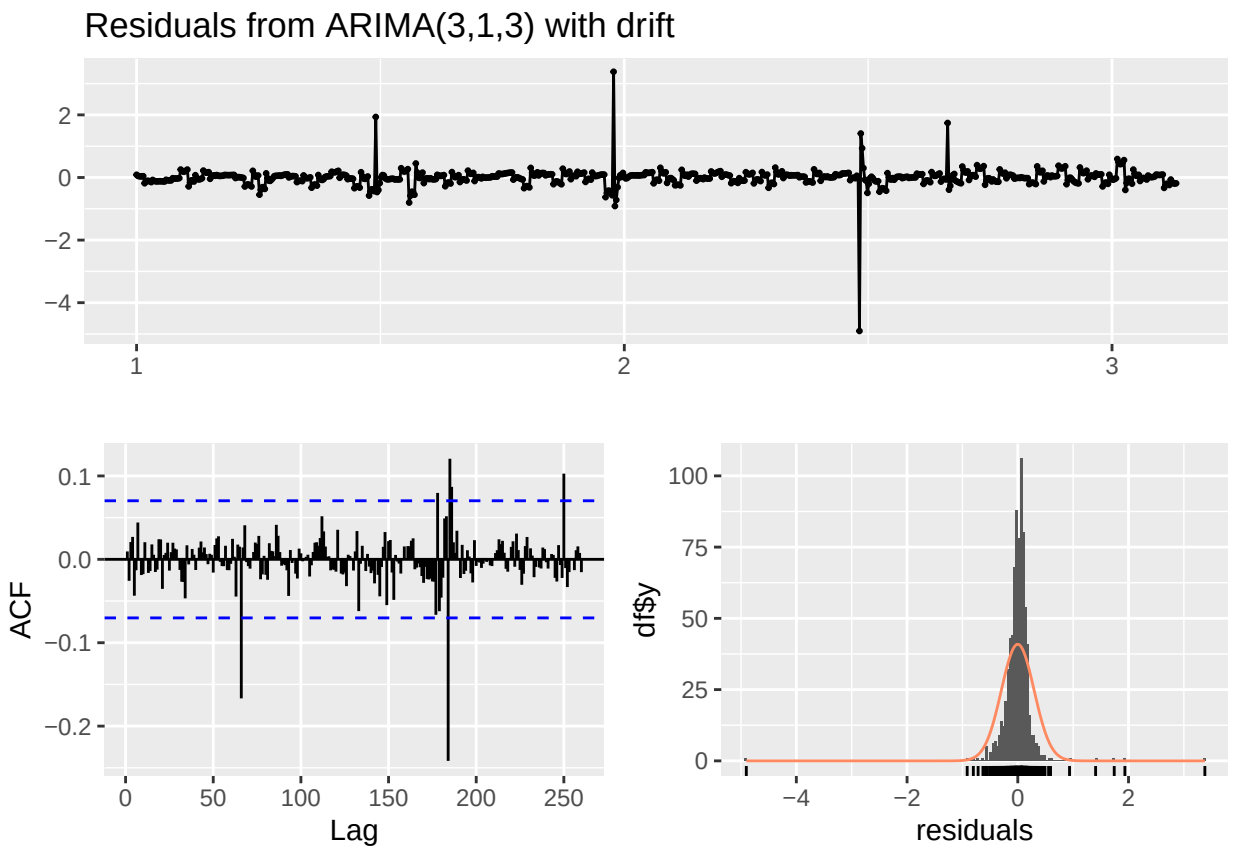
```
## Warning in adf.test(diff_data): p-value smaller than printed p-value
```

```
print(adf_test_diff)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_data
```

```
## Dickey-Fuller = -10.774, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```
fit <- auto.arima(ts_data, seasonal = FALSE)

checkresiduals(fit)
```

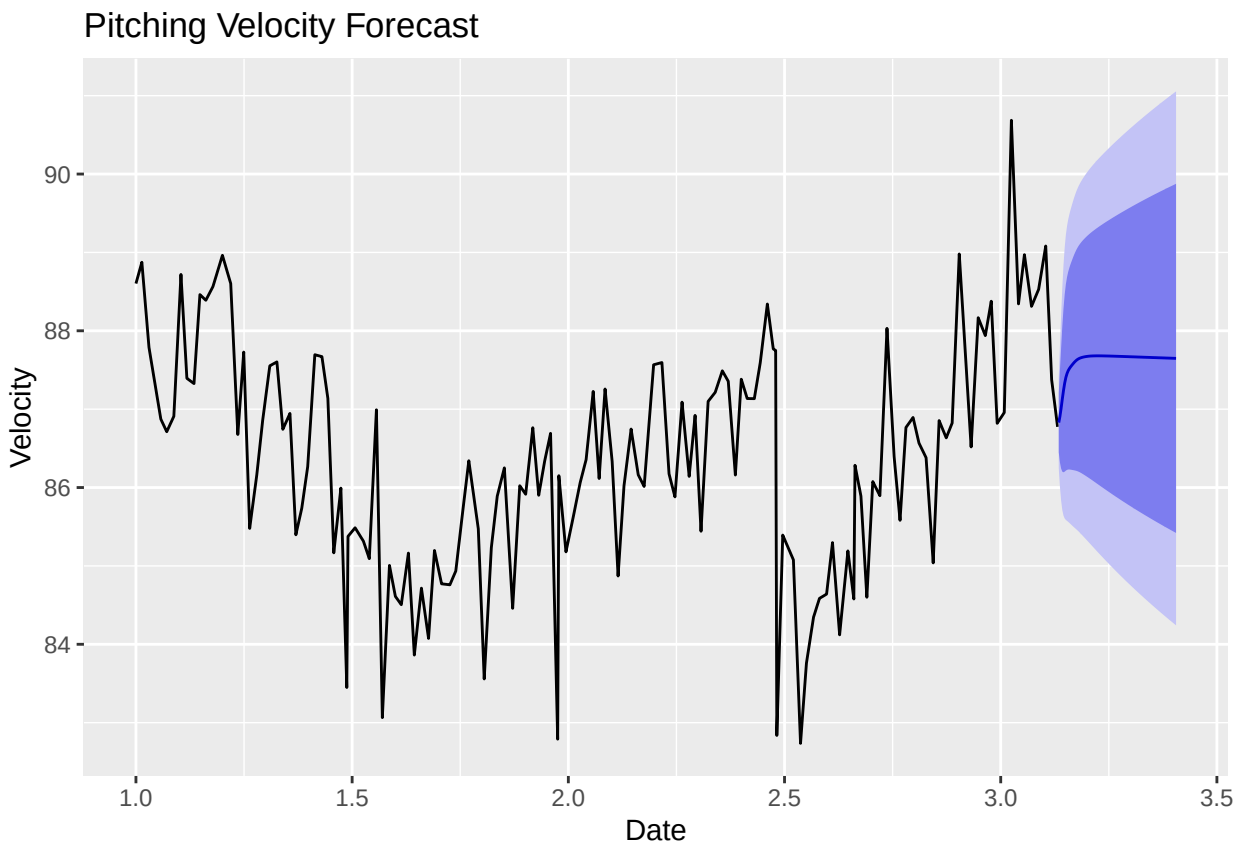



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,1,3) with drift
## Q* = 79.63, df = 150, p-value = 1
##
## Model df: 6.   Total lags used: 156
```

```
predictions <- forecast(fit, h = 100)

print(fit)
```

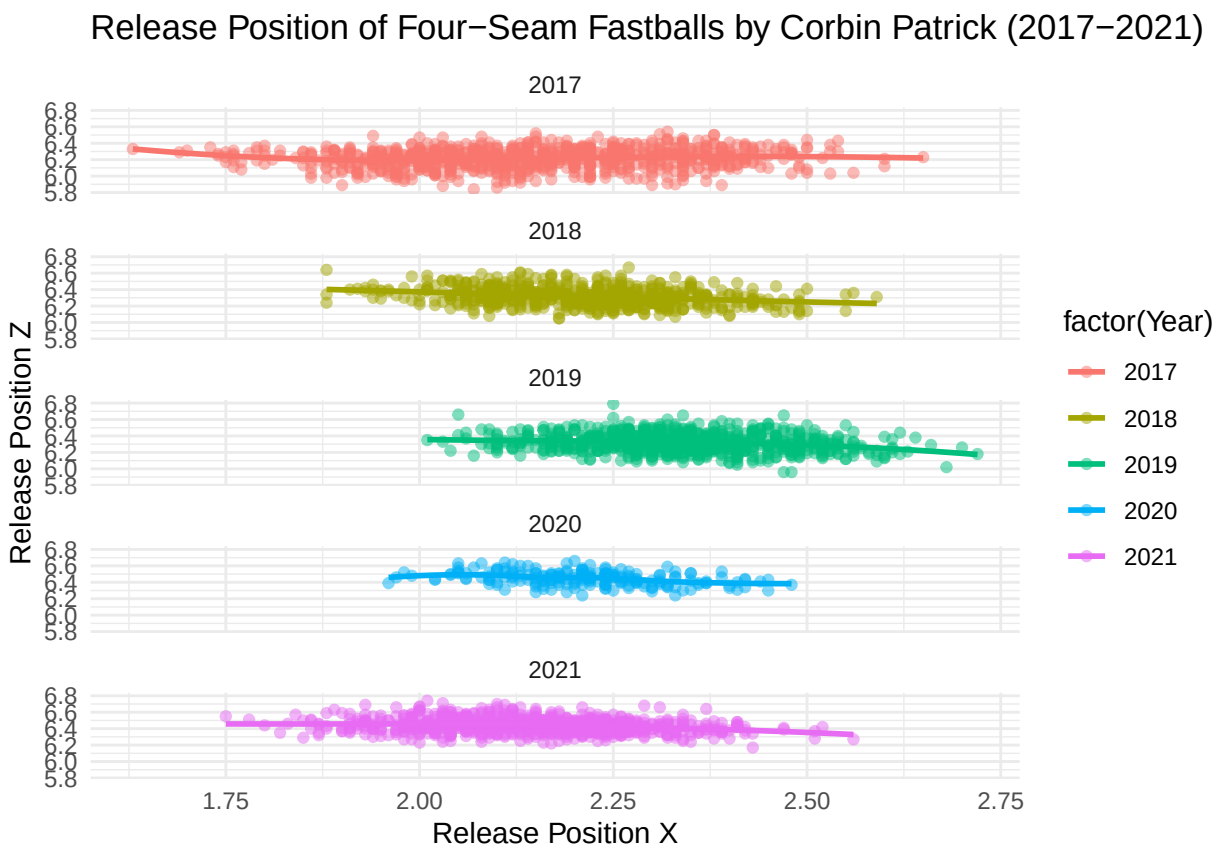
```
## Series: ts_data
## ARIMA(3,1,3) with drift
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2      ma3    drift
##       2.1023  -1.6394  0.4874  -1.9207  1.3941  -0.4492  -0.0005
## s.e.  0.2020   0.3203  0.1425   0.1961  0.2709   0.1031   0.0053
##
## sigma^2 = 0.08808:  log likelihood = -155.54
## AIC=327.08   AICc=327.26   BIC=364.33
```

```
autoplot(predictions) +
  labs(title = "Pitching Velocity Forecast", x = "Date", y = "Velocity")
```



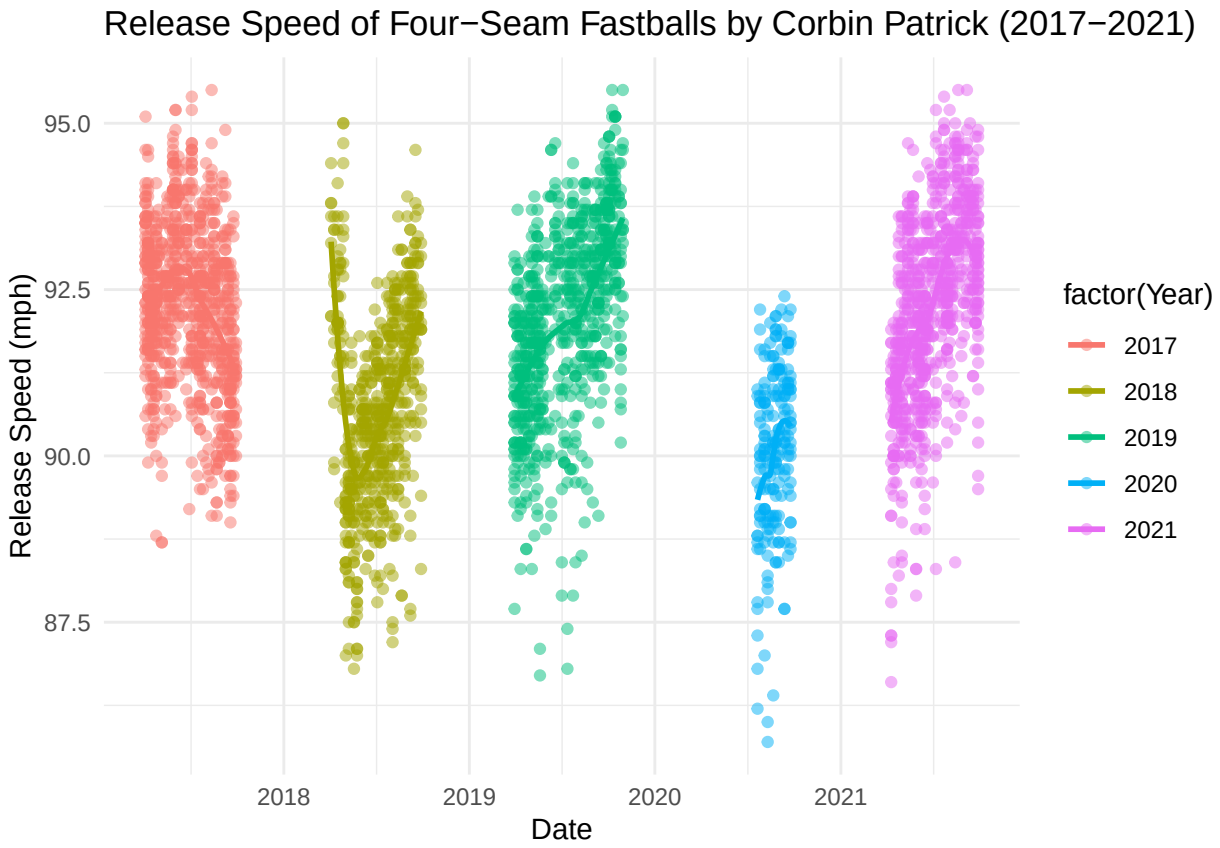

```
player %>%
  filter(player_name == "Corbin, Patrick", pitch_type == "FF") %>%
  ggplot(aes(x = release_pos_x, y = release_pos_z, color = factor(Year))) +
  geom_point(alpha = 0.5) +
  stat_smooth(aes(group = Year), method = 'loess', se = FALSE) +
  facet_wrap(~ Year, ncol = 1) +
  ggtitle("Release Position of Four-Seam Fastballs by Corbin Patrick (2017-2021)") +
  xlab("Release Position X") +
  ylab("Release Position Z") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
player %>%
  filter(player_name == "Corbin, Patrick", pitch_type == "FF") %>%
  ggplot(aes(x = as.Date(game_date), y = release_speed, color = factor(Year))) +
  geom_point(alpha = 0.5) +
  stat_smooth(aes(group = Year), method = 'loess', se = FALSE) +
  ggtitle("Release Speed of Four-Seam Fastballs by Corbin Patrick (2017-2021)") +
  xlab("Date") +
  ylab("Release Speed (mph)") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
numeric_vars <- player %>%
  select(release_speed, pfx_x, pfx_z, plate_x, plate_z,
         vx0, vy0, vz0, ax, ay, az, release_pos_x, release_pos_z, release_spin_rate, )


cor_matrix <- cor(numeric_vars)

print(cor_matrix)
```

```
##                  release_speed       pfx_x      pfx_z     plate_x     plate_z
## release_speed        1.00000000  0.76205550  0.7796455  0.10352536  0.35704436
## pfx_x                0.76205550  1.00000000  0.7938000  0.40960062  0.33137883
## pfx_z                0.77964547  0.79379997  1.0000000  0.16730009  0.48758609
## plate_x              0.10352536  0.40960062  0.1673001  1.00000000  0.25450921
## plate_z              0.35704436  0.33137883  0.4875861  0.25450921  1.00000000
## vx0                 -0.55329670 -0.39301755 -0.4747607  0.65728310 -0.03915815
## vy0                 -0.99983701 -0.76154011 -0.7779487 -0.11172040 -0.36405697
## vz0                 -0.58376554 -0.50914815 -0.4874734  0.08672871  0.46864140
## ax                   0.79567310  0.99406466  0.7965078  0.36621117  0.33417273
## ay                   0.86438868  0.69315714  0.7082218  0.13472188  0.30767535
## az                   0.82953010  0.80179337  0.9919554  0.14741818  0.45107329
## release_pos_x       -0.42753512 -0.29969980 -0.4234389  0.08104784 -0.18612964
## release_pos_z       -0.07951595  0.07692166  0.1791707  0.01657244  0.23488382
## release_spin_rate            NA          NA         NA          NA          NA
##                           vx0         vy0         vz0          ax          ay
## release_speed     -0.55329670 -0.99983701 -0.58376554  0.79567310  0.86438868
## pfx_x             -0.39301755 -0.76154011 -0.50914815  0.99406466  0.69315714
```

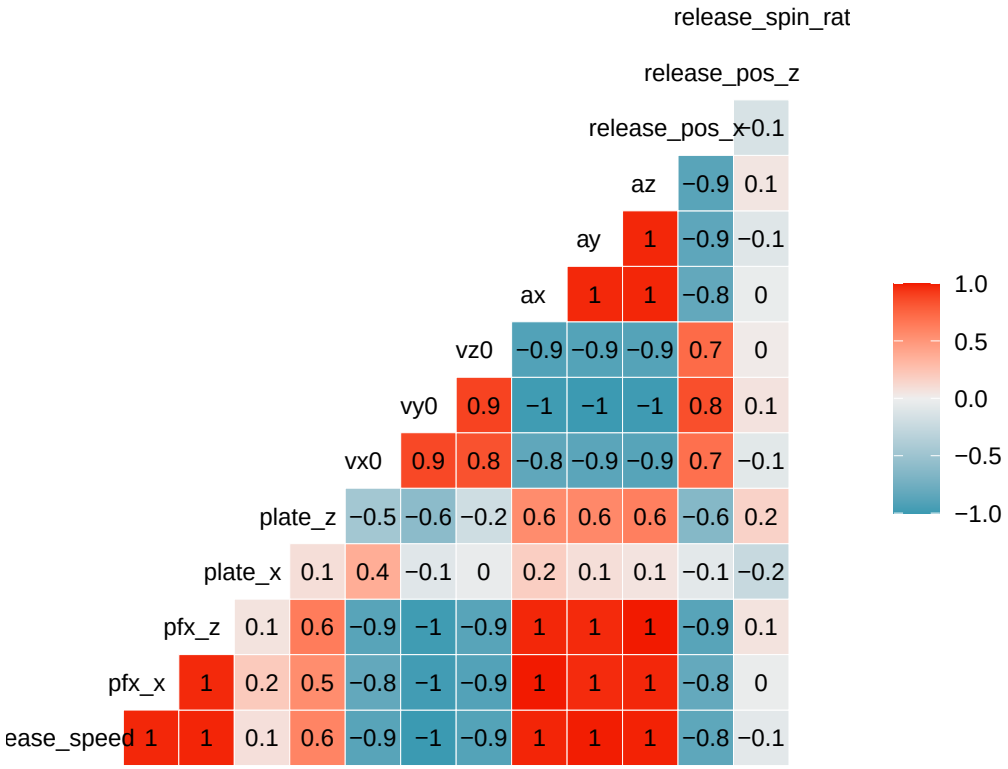
```
## pfx_z             -0.47476068 -0.77794868 -0.48747339  0.79650781  0.70822182
## plate_x            0.65728310 -0.11172040  0.08672871  0.36621117  0.13472188
## plate_z           -0.03915815 -0.36405697  0.46864140  0.33417273  0.30767535
## vx0                1.00000000  0.54505388  0.49241765 -0.43804591 -0.45203010
## vy0                0.54505388  1.00000000  0.57672421 -0.79471475 -0.86087213
## vz0                0.49241765  0.57672421  1.00000000 -0.52467051 -0.51062084
## ax                -0.43804591 -0.79471475 -0.52467051  1.00000000  0.73140867
## ay                -0.45203010 -0.86087213 -0.51062084  0.73140867  1.00000000
## az                -0.50664086 -0.82748356 -0.54272295  0.81352236  0.75468605
## release_pos_x      0.18517660  0.42665523  0.28363381 -0.31306841 -0.39529831
## release_pos_z     -0.01360712  0.07755964  0.02830721  0.07102084 -0.07366002
## release_spin_rate          NA          NA          NA          NA          NA
##                           az release_pos_x release_pos_z release_spin_rate
## release_speed      0.8295301   -0.42753512   -0.07951595                NA
## pfx_x              0.8017934   -0.29969980    0.07692166                NA
## pfx_z              0.9919554   -0.42343892    0.17917070                NA
## plate_x            0.1474182    0.08104784    0.01657244                NA
## plate_z            0.4510733   -0.18612964    0.23488382                NA
## vx0               -0.5066409    0.18517660   -0.01360712                NA
## vy0               -0.8274836    0.42665523    0.07755964                NA
## vz0               -0.5427230    0.28363381    0.02830721                NA
## ax                 0.8135224   -0.31306841    0.07102084                NA
## ay                 0.7546861   -0.39529831   -0.07366002                NA
## az                 1.0000000   -0.44591270    0.16024049                NA
## release_pos_x     -0.4459127    1.00000000   -0.07512110                NA
## release_pos_z      0.1602405   -0.07512110    1.00000000                NA
## release_spin_rate         NA            NA            NA                 1
```

```
library(GGally)
```

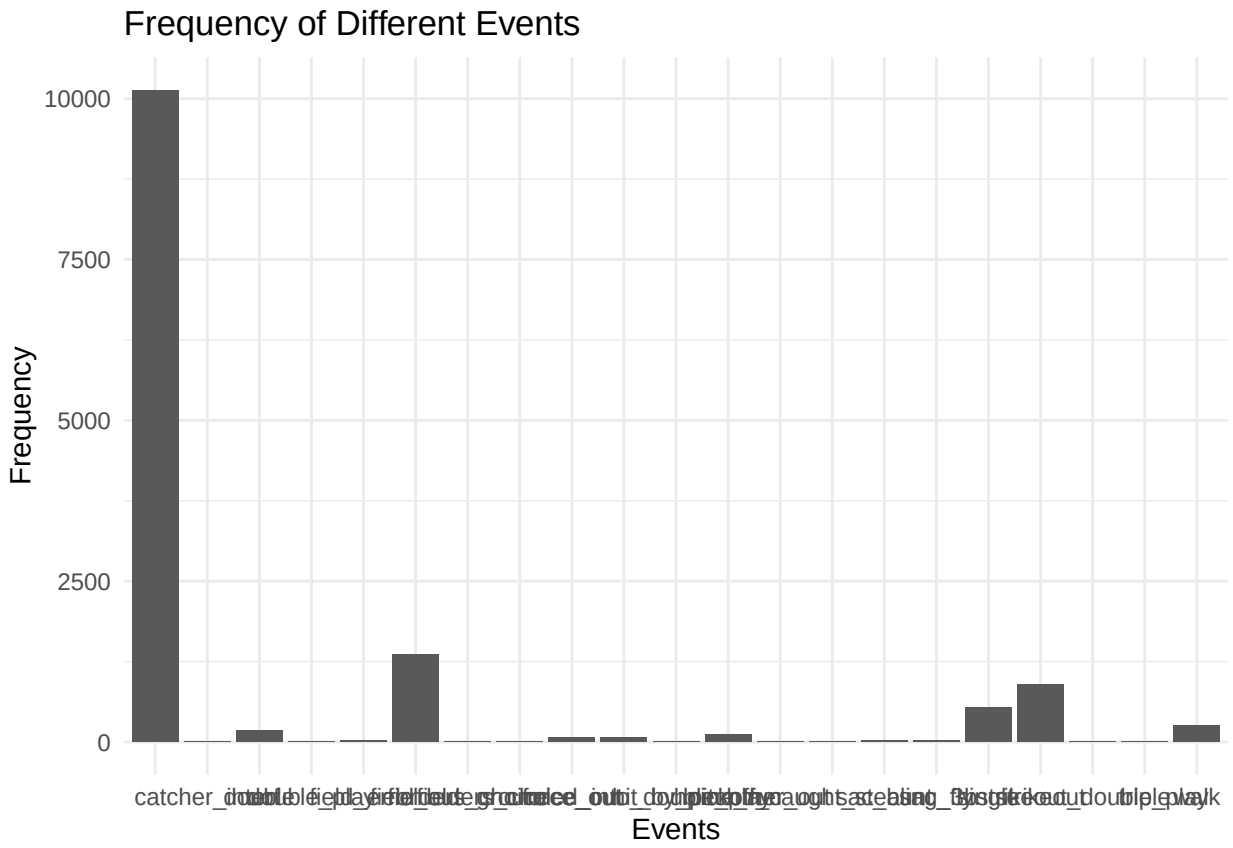
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggcorr(cor_matrix, label = TRUE, label_size = 3, hjust = 0.75, size = 3, layout.exp = 1) +
  ggtitle("Correlation Matrix of Pitching Variables") +
  theme_minimal()
```

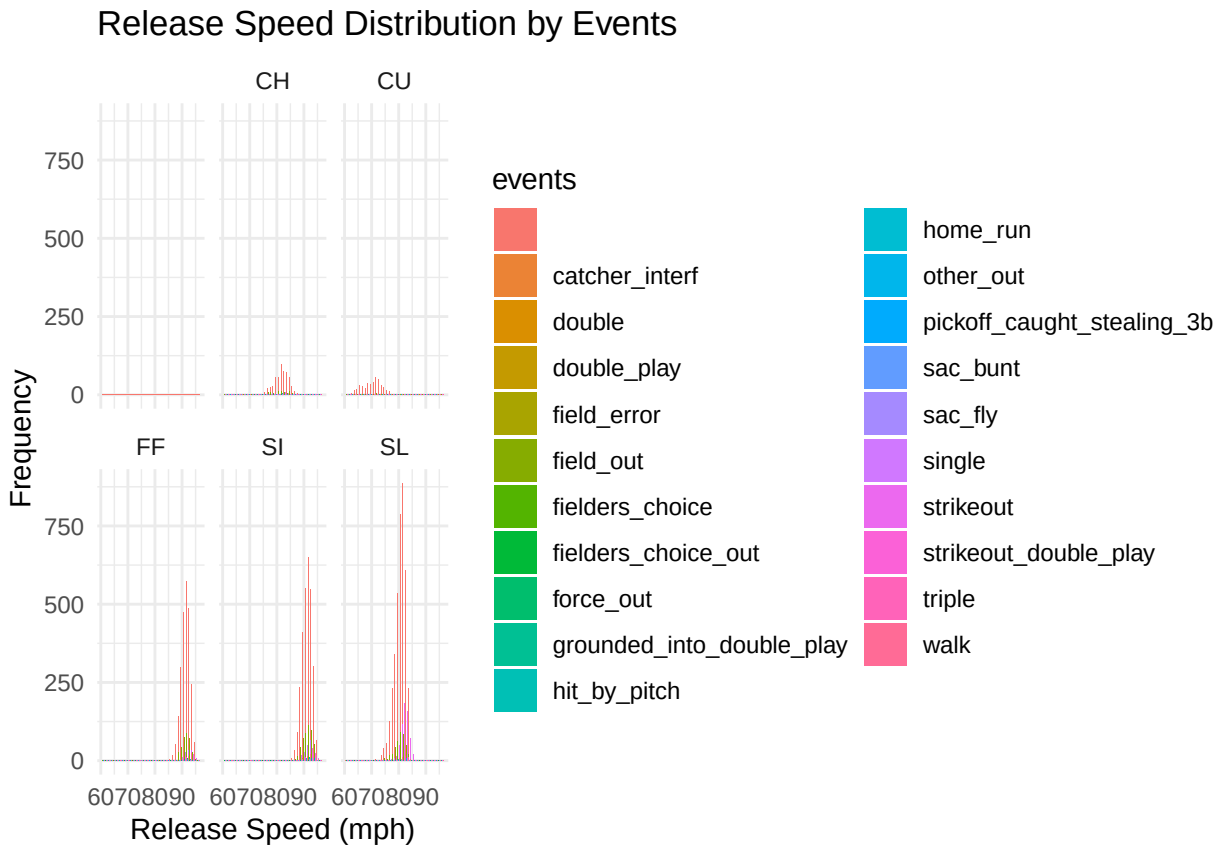
Correlation Matrix of Pitching Variables



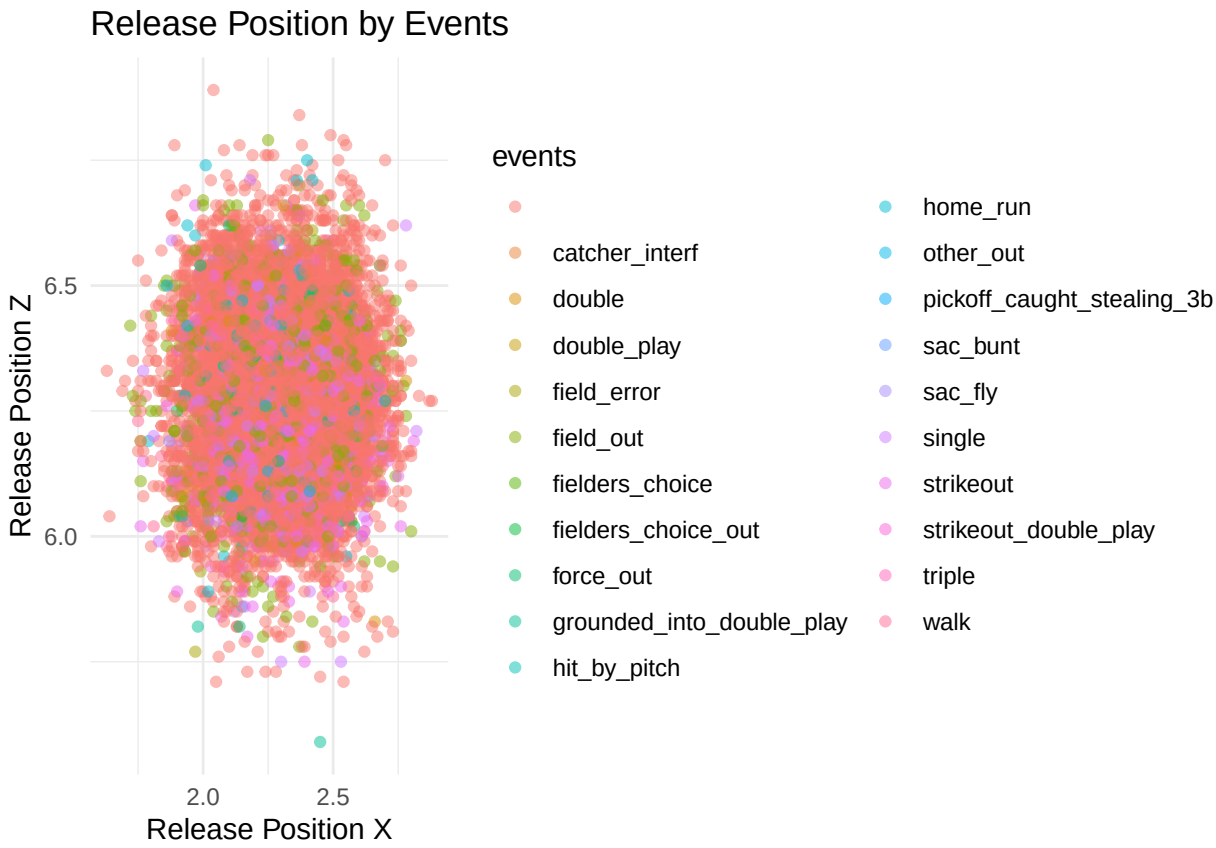

```r
library(tidyverse)

player_events <- player %>% filter(!is.na(events))

ggplot(player_events, aes(x = events)) +
  geom_bar() +
  theme_minimal() +
  ggtitle("Frequency of Different Events") +
  xlab("Events") +
  ylab("Frequency")
```



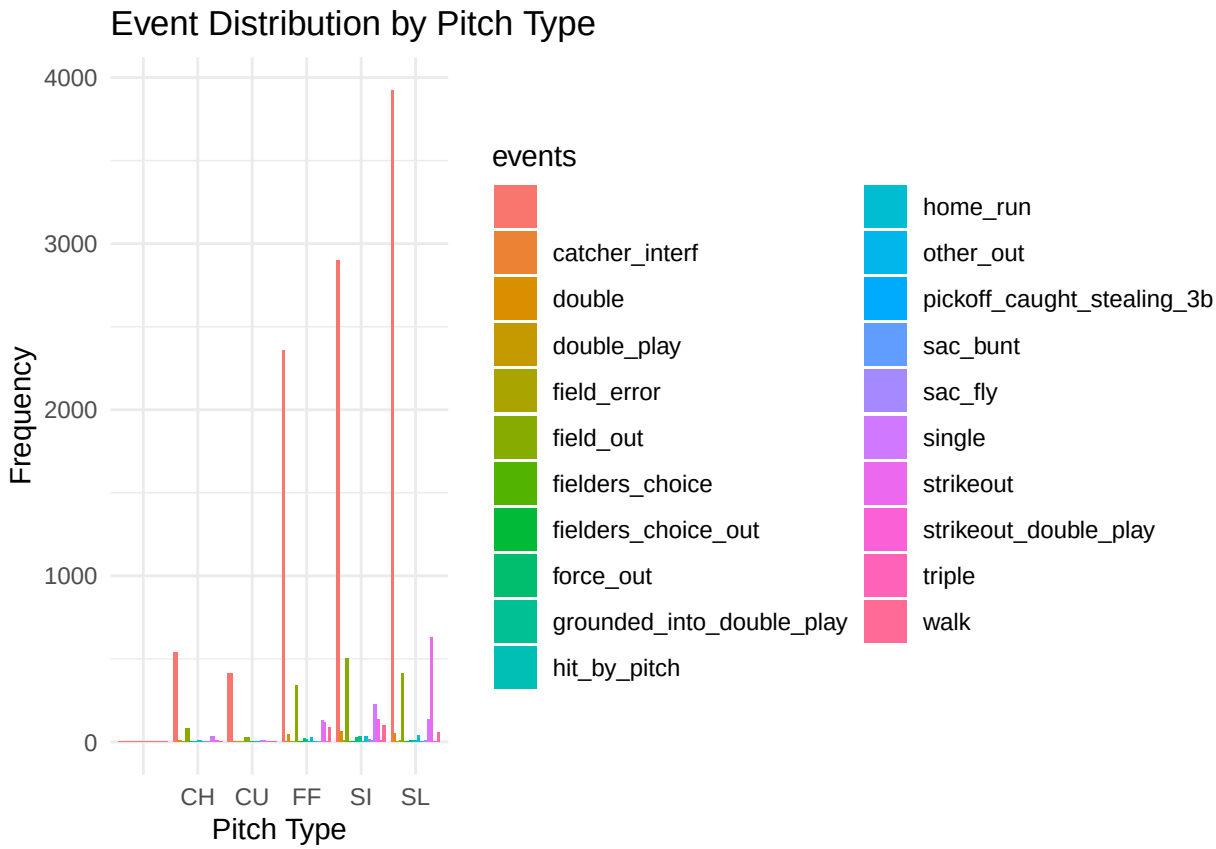

```
ggplot(player_events, aes(x = release_speed, fill = events)) +
  geom_histogram(binwidth = 1, position = "dodge") +
  theme_minimal() +
  ggtitle("Release Speed Distribution by Events") +
  xlab("Release Speed (mph)") +
  ylab("Frequency") +
  facet_wrap(~ pitch_type)
```



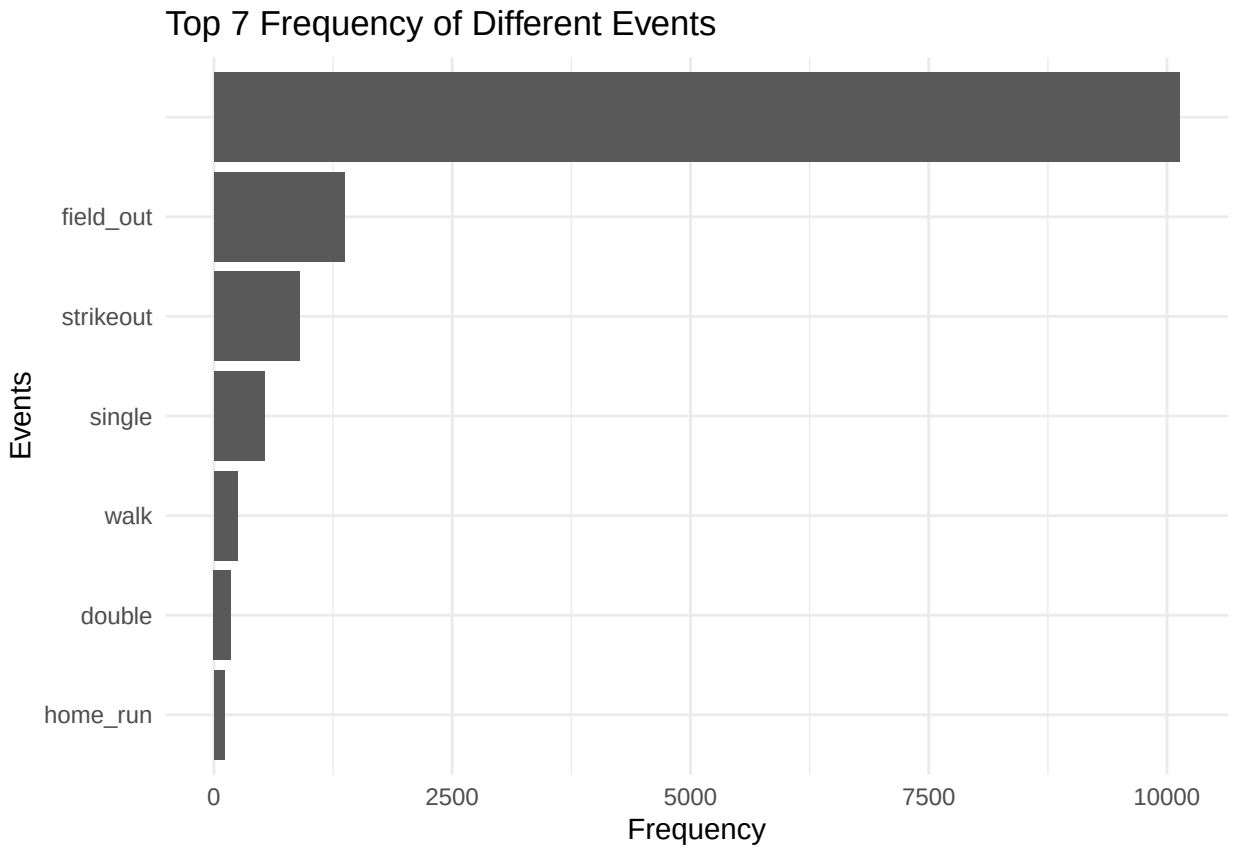

```
ggplot(player_events, aes(x = release_pos_x, y = release_pos_z, color = events)) +
  geom_point(alpha = 0.5) +
  theme_minimal() +
  ggtitle("Release Position by Events") +
  xlab("Release Position X") +
  ylab("Release Position Z")
```



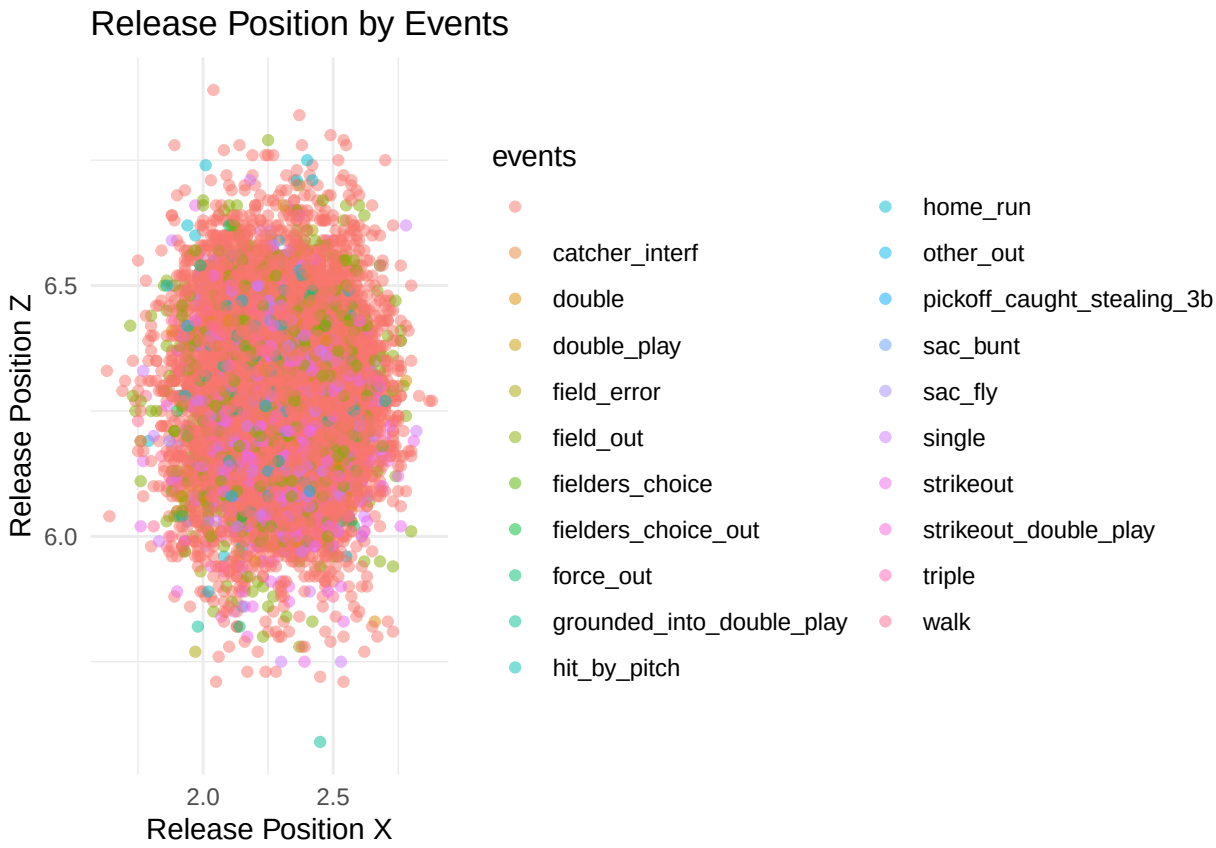

```r
ggplot(player_events, aes(x = pitch_type, fill = events)) +
  geom_bar(position = "dodge") +
  theme_minimal() +
  ggtitle("Event Distribution by Pitch Type") +
  xlab("Pitch Type") +
  ylab("Frequency")
```




```r
library(tidyverse)


player_events <- player %>% filter(!is.na(events))

top_events <- player_events %>%
  group_by(events) %>%
  summarise(frequency = n()) %>%
  arrange(desc(frequency)) %>%
  slice_head(n = 7)

ggplot(top_events, aes(x = reorder(events, frequency), y = frequency)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  ggtitle("Top 7 Frequency of Different Events") +
  xlab("Events") +
  ylab("Frequency") +
  coord_flip()
```




```r
player_events <- player %>% filter(!is.na(events))

player_events <- player_events %>% filter(!is.na(release_pos_x) & !is.na(release_pos_z) & !is.na(events)

ggplot(player_events, aes(x = release_pos_x, y = release_pos_z, color = events)) +
  geom_point(alpha = 0.5) +
  theme_minimal() +
  ggtitle("Release Position by Events") +
  xlab("Release Position X") +
  ylab("Release Position Z")
```




```r
important_vars <- player %>%
  select(pitch_type, release_speed, release_pos_x, release_pos_z, events, description)

summary_table <- important_vars %>%
  summarise(
    pitch_type_count = n_distinct(pitch_type),
    avg_release_speed = mean(release_speed, na.rm = TRUE),
    sd_release_speed = sd(release_speed, na.rm = TRUE),
    avg_release_pos_x = mean(release_pos_x, na.rm = TRUE),
    sd_release_pos_x = sd(release_pos_x, na.rm = TRUE),
    avg_release_pos_z = mean(release_pos_z, na.rm = TRUE),
    sd_release_pos_z = sd(release_pos_z, na.rm = TRUE),
    event_count = n_distinct(events),
    description_count = n_distinct(description)
  )

write.csv(summary_table, "summary_table.csv", row.names = FALSE)
```

```r
player_events <- player %>%
  filter(!is.na(events))

selected_events <- c("field_out", "strikeout", "single", "walk", "double", "home_run")

player_events <- player_events %>%
  filter(events %in% selected_events)

player_events$events <- factor(player_events$events, levels = selected_events)
```

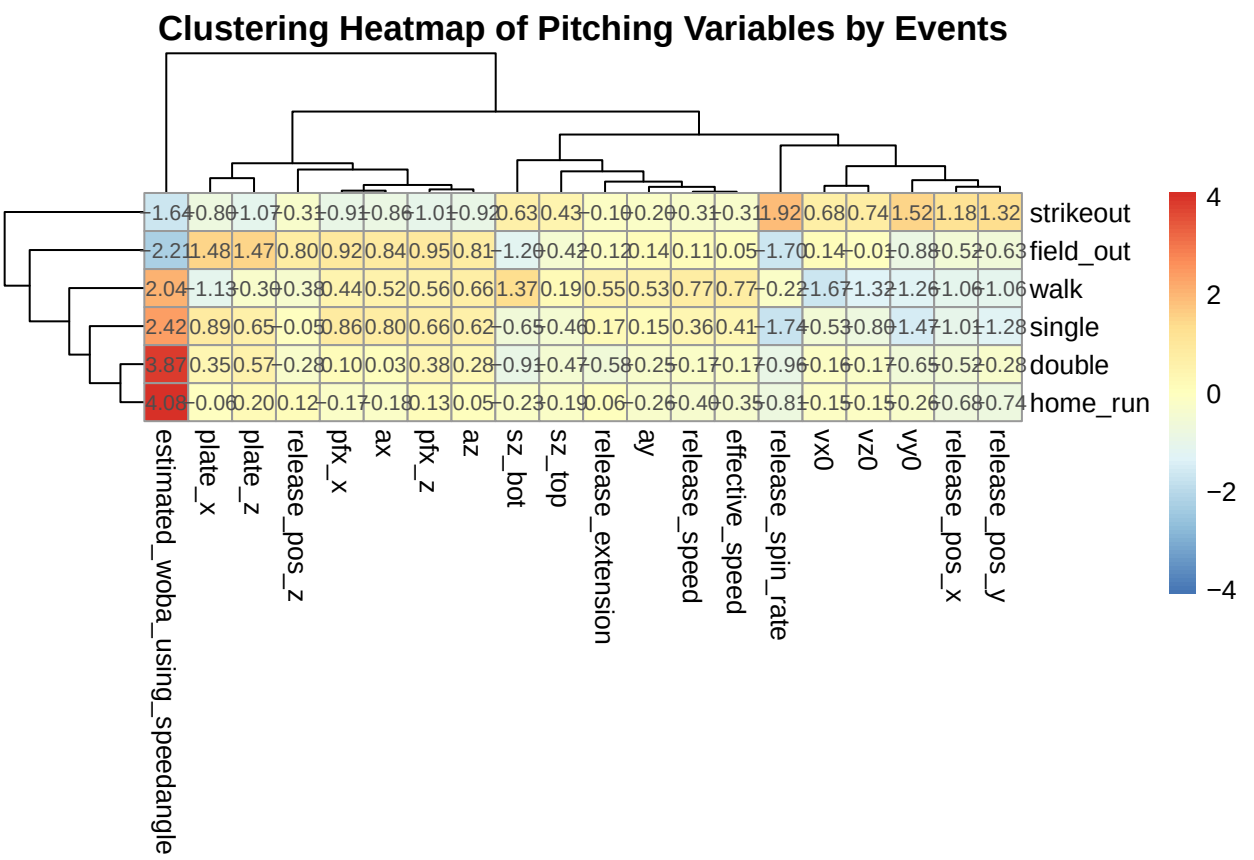
```r
numeric_vars <- player_events %>%
  select_if(is.numeric) %>%
  colnames()

for (var in numeric_vars) {
  p <- ggplot(player_events, aes_string(x = "events", y = var, fill = "events")) +
    geom_violin() +
    theme_minimal() +
    ggtitle(paste("Violin Plot of", var, "by Events")) +
    xlab("Events") +
    ylab(var)
}
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
selected_events <- c("field_out", "strikeout", "single", "walk", "double", "home_run")

player_events <- player_events %>%
  filter(events %in% selected_events)

numeric_vars <- c("release_speed", "release_pos_x", "release_pos_z", "pfx_x", "pfx_z",
                  "plate_x", "plate_z", "vx0", "vy0", "vz0", "ax", "ay", "az",
                  "sz_top", "sz_bot", "effective_speed", "release_spin_rate",
                  "release_extension", "release_pos_y", "estimated_woba_using_speedangle")

data_numeric <- player_events %>%
  select(all_of(numeric_vars)) %>%
  mutate(across(everything(), ~ scale(.)))

data_numeric <- cbind(data_numeric, events = player_events$events)

data_means <- data_numeric %>%
  group_by(events) %>%
  summarise(across(everything(), mean, na.rm = TRUE))
```

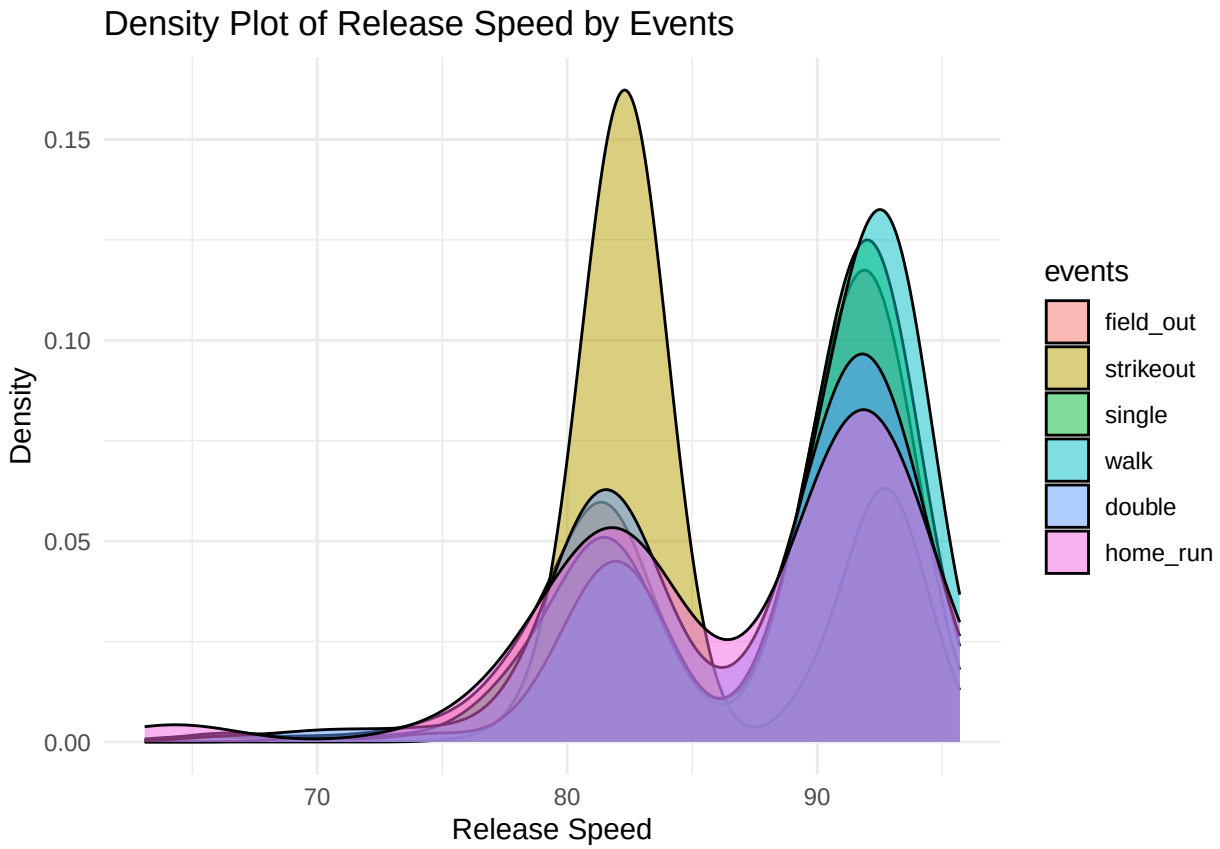
```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(everything(), mean, na.rm = TRUE)`.
## i In group 1: `events = field_out`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```
data_matrix <- as.matrix(data_means[,-1])

rownames(data_matrix) <- data_means$events

pheatmap(data_matrix,
         scale = "row",
         clustering_distance_rows = "euclidean",
         clustering_distance_cols = "euclidean",
         clustering_method = "complete",
         display_numbers = TRUE,
         main = "Clustering Heatmap of Pitching Variables by Events")
```




```
data_numeric <- player_events %>%
  select(all_of(numeric_vars))

cor_matrix <- cor(data_numeric, use = "complete.obs")
```

```
ggplot(player_events, aes(x = release_speed, fill = events)) +
  geom_density(alpha = 0.5) +
  theme_minimal() +
  ggtitle("Density Plot of Release Speed by Events") +
  xlab("Release Speed") +
  ylab("Density")
```

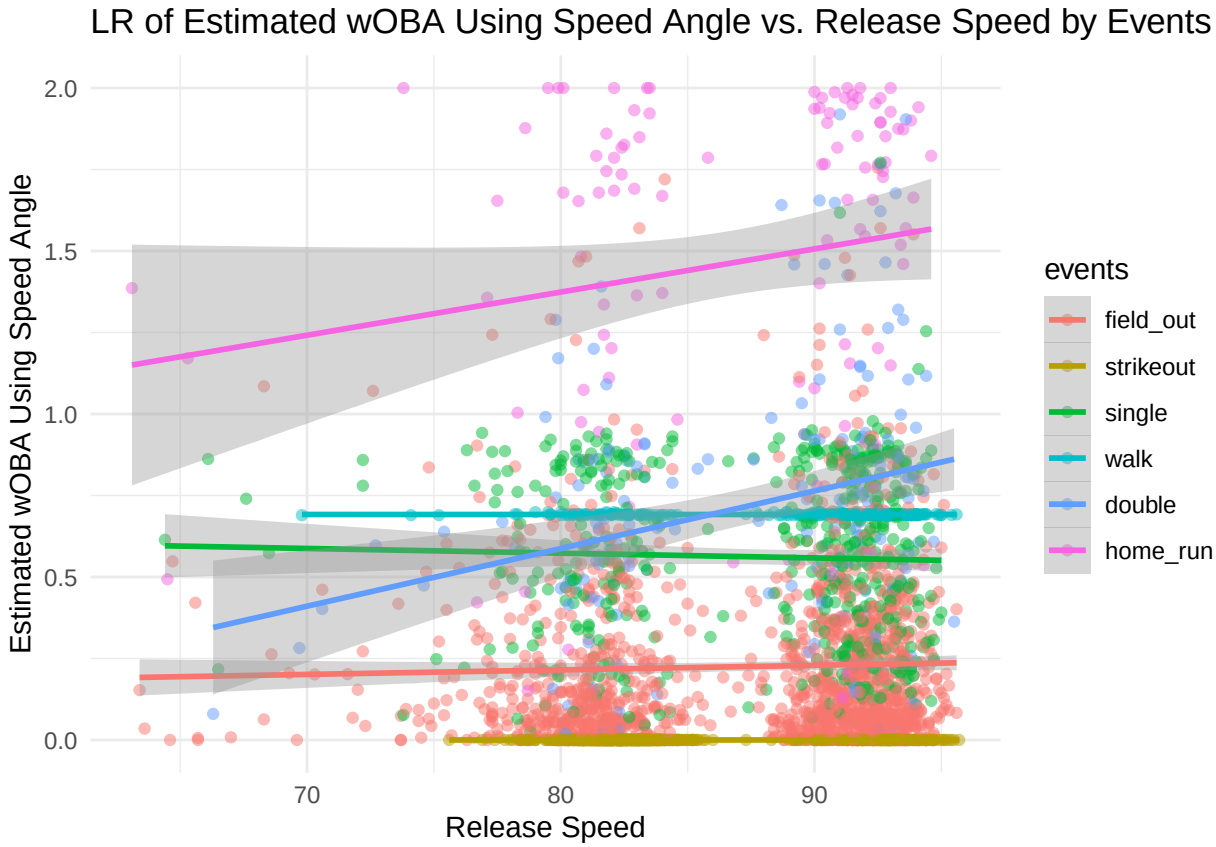



```
ggplot(player_events, aes(x = release_speed, y = estimated_woba_using_speedangle, color = events)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  theme_minimal() +
  ggtitle("LR of Estimated wOBA Using Speed Angle vs. Release Speed by Events") +
  xlab("Release Speed") +
  ylab("Estimated wOBA Using Speed Angle")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 29 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 29 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

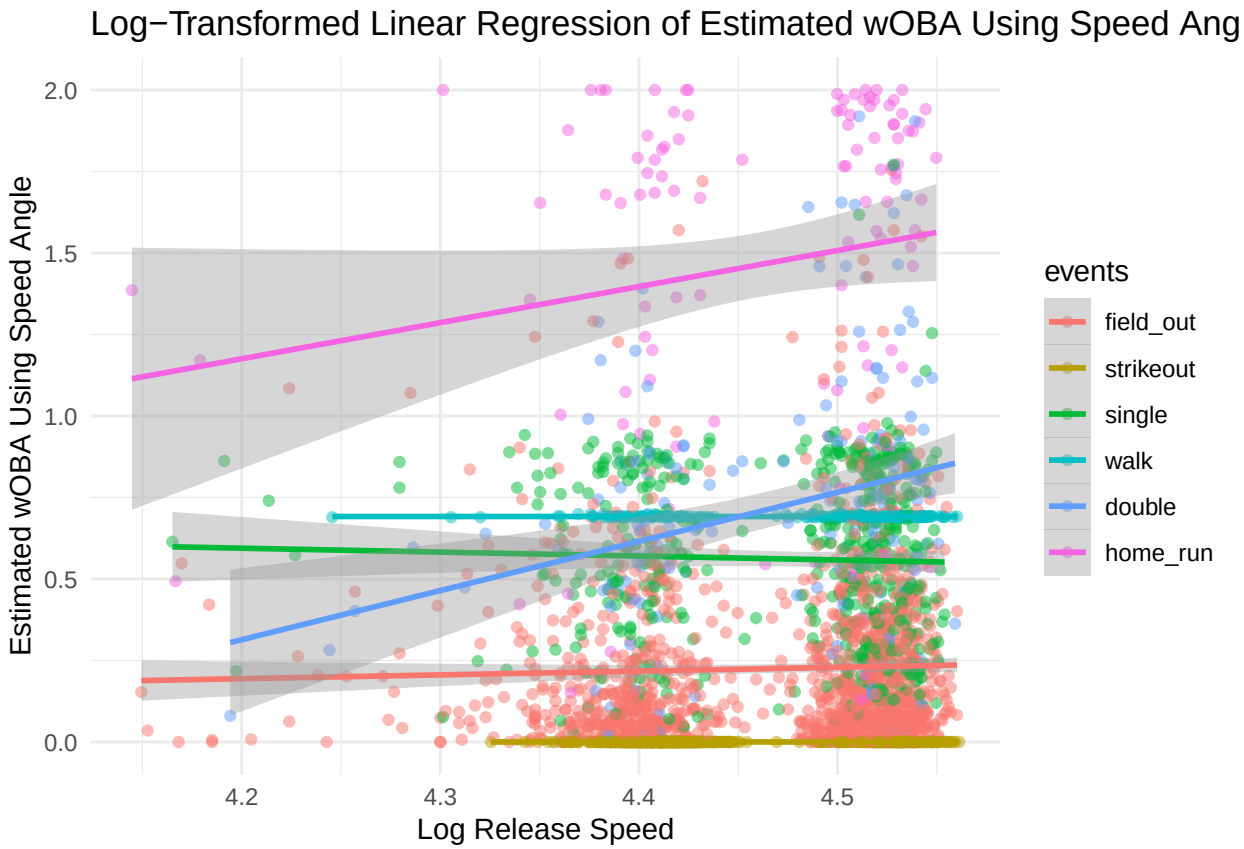



```
player_events <- player_events %>%
  mutate(log_release_speed = log(release_speed))

ggplot(player_events, aes(x = log_release_speed, y = estimated_woba_using_speedangle, color = events)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  theme_minimal() +
  ggtitle("Log-Transformed Linear Regression of Estimated wOBA Using Speed Angle vs. Log Release Speed") +
  xlab("Log Release Speed") +
  ylab("Estimated wOBA Using Speed Angle")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 29 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

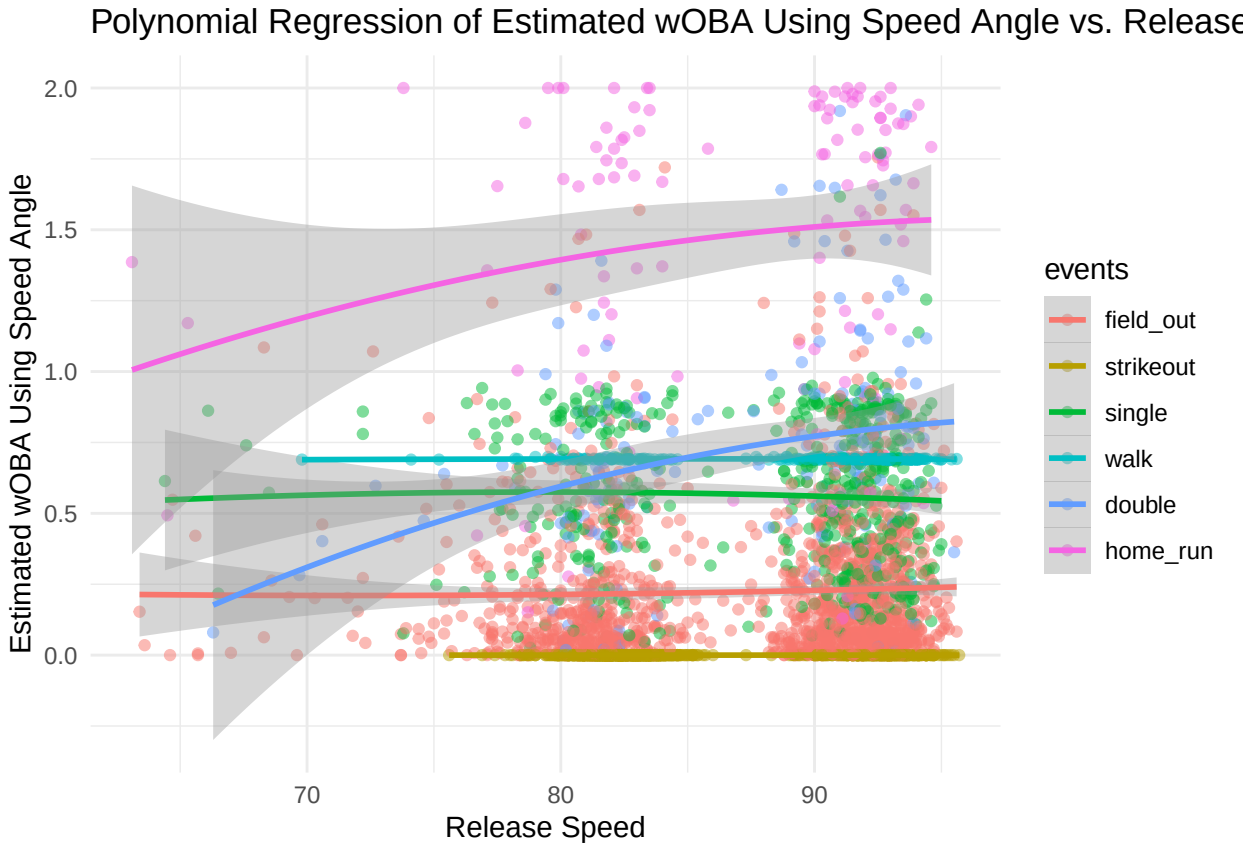
```
## Warning: Removed 29 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
ggplot(player_events, aes(x = release_speed, y = estimated_woba_using_speedangle, color = events)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", formula = y ~ poly(x, 2)) +
  theme_minimal() +
  ggtitle("Polynomial Regression of Estimated wOBA Using Speed Angle vs. Release Speed by Events") +
  xlab("Release Speed") +
  ylab("Estimated wOBA Using Speed Angle")
```

```
## Warning: Removed 29 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 29 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
player_events <- player %>%
  filter(!is.na(events))

selected_events <- c("field_out", "strikeout", "single", "walk", "double", "home_run")

player_events <- player_events %>%
  filter(events %in% selected_events)

numeric_vars <- c("release_speed", "release_pos_x", "release_pos_z", "pfx_x", "pfx_z",
                  "plate_x", "plate_z", "vx0", "vy0", "vz0", "ax", "ay", "az",
                  "sz_top", "sz_bot", "effective_speed", "release_spin_rate",
                  "release_extension", "release_pos_y", "estimated_woba_using_speedangle")

player_events <- player_events %>%
  select(all_of(numeric_vars), events) %>%
  drop_na()

X <- model.matrix(estimated_woba_using_speedangle ~ release_speed + release_pos_x + release_pos_z +
                    pfx_x + pfx_z + plate_x + plate_z + vx0 + vy0 + vz0 +
                    ax + ay + az + sz_top + sz_bot + effective_speed +
                    release_spin_rate + release_extension + release_pos_y + events,
                  data = player_events)
y <- player_events$estimated_woba_using_speedangle

ridge_model <- cv.glmnet(X, y, alpha = 0)

best_lambda <- ridge_model$lambda.min
```

```r
player_events$predicted_woba <- predict(ridge_model, s = best_lambda, newx = X)

ggplot(player_events, aes(x = release_speed, y = predicted_woba, color = events)) +
  geom_point(alpha = 0.5) +
  theme_minimal() +
  ggtitle("Ridge Regression of EwOBA Using Speed Angle vs. Release Speed by Events") +
  xlab("Release Speed") +
  ylab("Predicted wOBA Using Speed Angle")
```