My title[*]

My subtitle if needed

First author Another author

November 29, 2024

First sentence. Second sentence. Third sentence. Fourth sentence.

1 Introduction

[*]Code and data are available at: https://github.com/RohanAlexander/starter_folder.

2 Data

```r
cleaned_data %>% select(!c(ball, innings, match_id)) %>% head()
```

```
# A tibble: 6 x 17
   year venue         over batting_team bowling_team striker bowler runs_off_bat
  <dbl> <chr>        <dbl> <chr>        <chr>        <chr>   <chr>         <dbl>
1  2021 MA Chidamba~     1 Mumbai Indi~ Royal Chall~ RG Sha~ Moham~            0
2  2021 MA Chidamba~     1 Mumbai Indi~ Royal Chall~ RG Sha~ Moham~            0
3  2021 MA Chidamba~     1 Mumbai Indi~ Royal Chall~ RG Sha~ Moham~            2
4  2021 MA Chidamba~     1 Mumbai Indi~ Royal Chall~ RG Sha~ Moham~            0
5  2021 MA Chidamba~     1 Mumbai Indi~ Royal Chall~ RG Sha~ Moham~            1
6  2021 MA Chidamba~     2 Mumbai Indi~ Royal Chall~ RG Sha~ KA Ja~            1
# i 9 more variables: wickets_lost_yet <dbl>, wicket <lgl>, target <dbl>,
#   run_rate <dbl>, batting_style <chr>, batter_playing_role <chr>,
#   bowling_style <chr>, bowler_playing_role <chr>, prev_over_wickets <int>
```

2.1 Measurement

2.2 Predictor Variables

```r
num_bowlers_per_type <- cleaned_data %>%
  group_by(bowling_style) %>%
  summarise(
    num_bowlers = n_distinct(bowler)
  )

num_batters_per_type <- cleaned_data %>%
  group_by(batting_style) %>%
  summarise(
    num_bowlers = n_distinct(striker)
  )
```

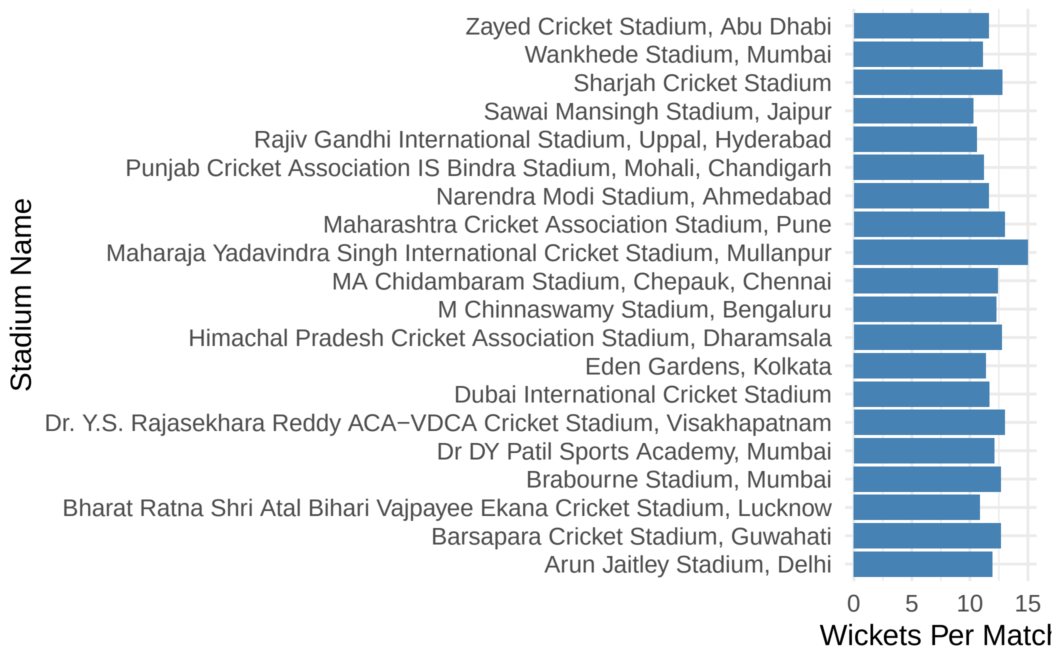
2.3 Data Cleaning

2.4 Relationship Between Wickets and other Variables

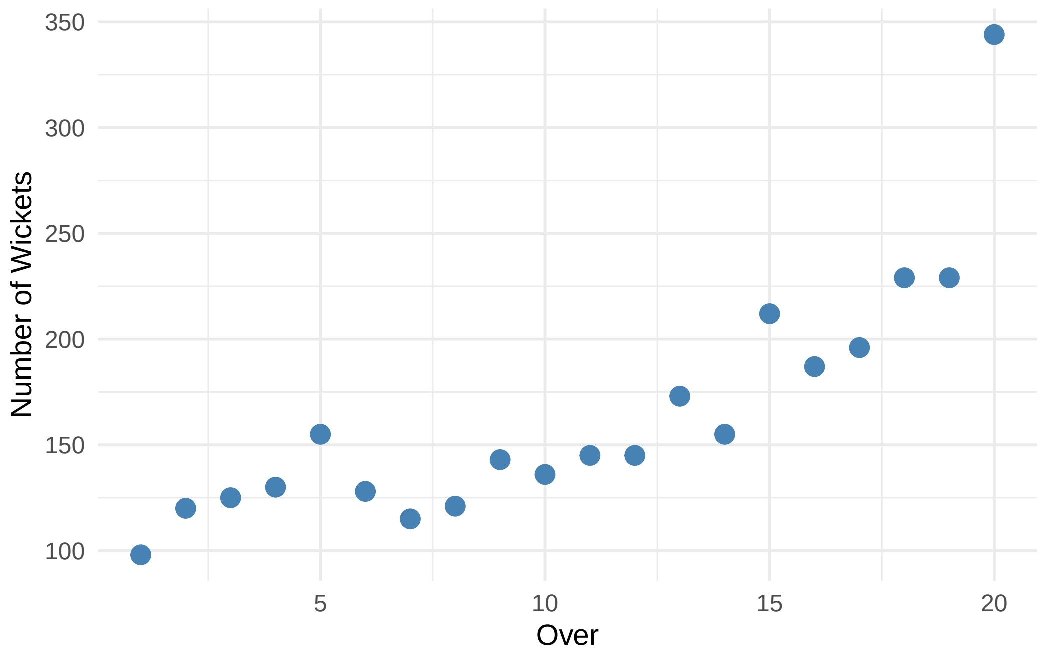
```r
cleaned_data %>% head()
```

```
# A tibble: 6 x 20
  match_id  year venue     innings  over  ball batting_team bowling_team striker
     <dbl> <dbl> <chr>       <dbl> <dbl> <dbl> <chr>        <chr>        <chr>
1  1254058  2021 MA Chida~       1     1     2 Mumbai Indi~ Royal Chall~ RG Sha~
2  1254058  2021 MA Chida~       1     1     3 Mumbai Indi~ Royal Chall~ RG Sha~
3  1254058  2021 MA Chida~       1     1     4 Mumbai Indi~ Royal Chall~ RG Sha~
4  1254058  2021 MA Chida~       1     1     5 Mumbai Indi~ Royal Chall~ RG Sha~
5  1254058  2021 MA Chida~       1     1     6 Mumbai Indi~ Royal Chall~ RG Sha~
6  1254058  2021 MA Chida~       1     2     1 Mumbai Indi~ Royal Chall~ RG Sha~
# i 11 more variables: bowler <chr>, runs_off_bat <dbl>,
#   wickets_lost_yet <dbl>, wicket <lgl>, target <dbl>, run_rate <dbl>,
#   batting_style <chr>, batter_playing_role <chr>, bowling_style <chr>,
#   bowler_playing_role <chr>, prev_over_wickets <int>
```

```r
stadium_boundaries <- cleaned_data %>%
  group_by(venue) %>%
  summarise(
    num_matches = n_distinct(match_id),
    num_wickets = sum(wicket == TRUE),
  ) %>% arrange(desc(num_wickets), desc(num_matches))

ggplot(stadium_boundaries, aes(x = venue, y = (num_wickets/num_matches))) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(
       x = "Stadium Name",
       y = "Wickets Per Match") +
  theme_minimal() +
  coord_flip()
```



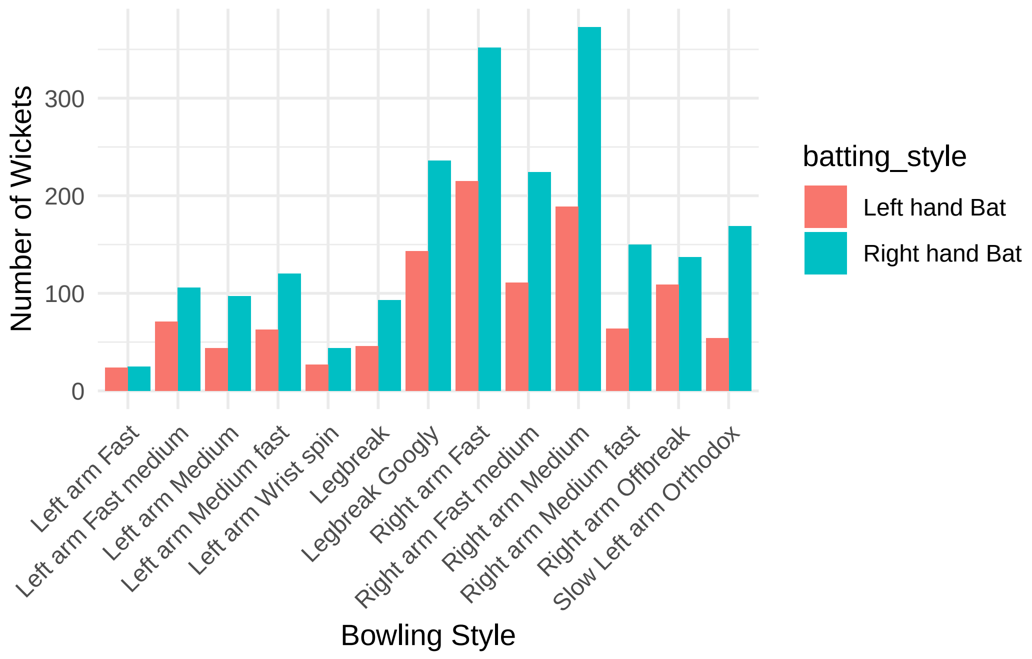
```r
over_boundaries <- cleaned_data %>%
  group_by(over) %>%
  summarise(
    num_wickets = sum(wicket == TRUE),
    num_balls = n()
  ) %>% arrange(desc(num_wickets), desc(num_balls))

ggplot(over_boundaries, aes(x = over, y = num_wickets)) +
  geom_point(color = "steelblue", size = 3) +
  labs(
       x = "Over",
       y = "Number of Wickets") +
  theme_minimal()
```



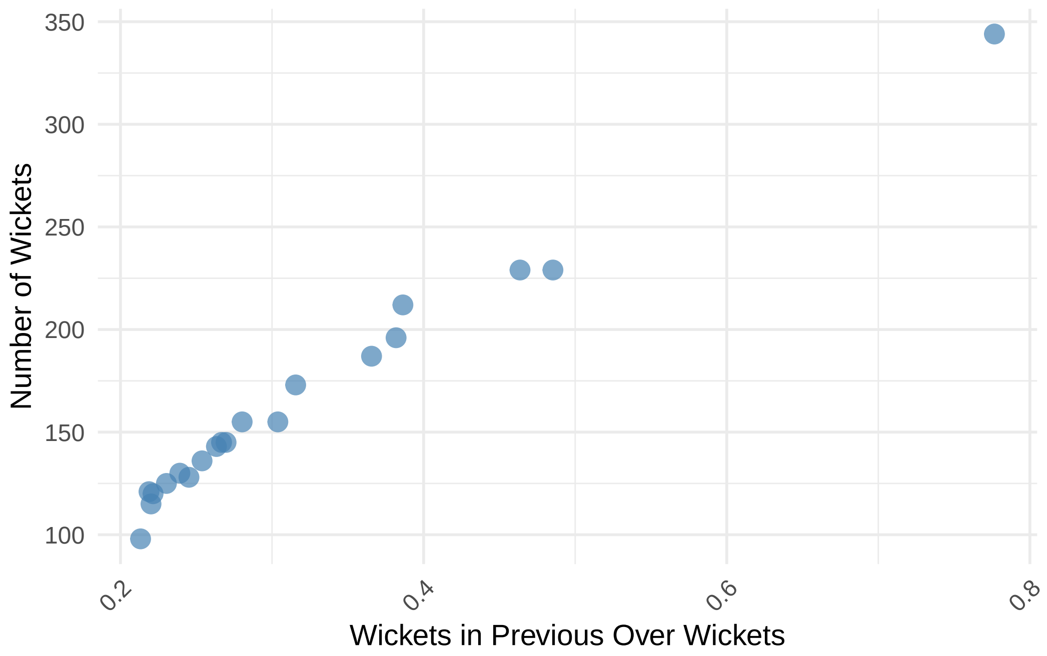
```r
bowling_batting_matchup_boundaries <- cleaned_data %>%
  group_by(bowling_style, batting_style) %>%
  summarise(
    num_wickets = sum(wicket == TRUE),
    num_balls = n(),
  ) %>% arrange(desc(num_wickets), desc(num_balls))
```

```
`summarise()` has grouped output by 'bowling_style'. You can override using the
`.groups` argument.
```

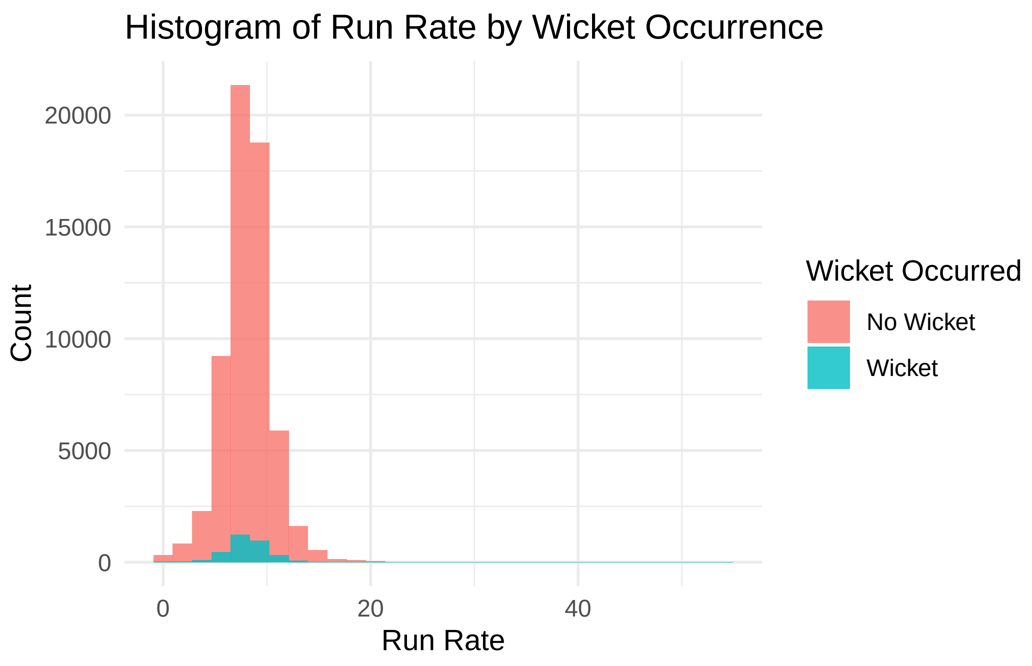
```r
ggplot(bowling_batting_matchup_boundaries, aes(x = bowling_style, y = num_wickets, fill = ba
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(
       x = "Bowling Style",
       y = "Number of Wickets") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
wickets_prev_over_wickets <- cleaned_data %>%
  group_by(over) %>%
  summarise(
    num_wickets = sum(wicket == TRUE),
    prev_over_wickets = mean(prev_over_wickets),
    num_balls = n(),
  ) %>% arrange(desc(num_wickets), desc(num_balls))

ggplot(wickets_prev_over_wickets, aes(x = prev_over_wickets, y = num_wickets)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.7) +
  labs(
       x = "Wickets in Previous Over Wickets",
       y = "Number of Wickets") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
ggplot(cleaned_data, aes(x = run_rate, fill = factor(wicket))) +
  geom_histogram(position = "identity", alpha = 0.8, bins = 30) +
  labs(
    x = "Run Rate",
    y = "Count",
    title = "Histogram of Run Rate by Wicket Occurrence"
  ) +
  scale_fill_discrete(name = "Wicket Occurred", labels = c("No Wicket", "Wicket")) +
  theme_minimal()
```



3 Model

3.1 Model set-up

3.1.1 Model justification

4 Results

```
simple_glm_wicket_model <- readRDS(here("models/simple_glm_wicket_model.rds"))

#summary(simple_glm_wicket_model)
modelsummary(simple_glm_wicket_model)
```

```
complex_glm_wicket_model <- readRDS(here("models/complex_glm_wicket_model.rds"))
#summary(complex_glm_wicket_model)
modelsummary(complex_glm_wicket_model)
```

	(1)
(Intercept)	−3.569
	(0.048)
over	0.058
	(0.004)
Num.Obs.	51 587
AIC	20 589.7
BIC	20 607.4
Log.Lik.	−10 292.858
RMSE	0.22

	(1)
(Intercept)	−4.139
	(0.052)
over	0.005
	(0.004)
prev_over_wickets	1.852
	(0.031)
Num.Obs.	51 587
AIC	16 590.1
BIC	16 616.7
Log.Lik.	−8292.074
RMSE	0.21

```
overly_complex_glm_wicket_model <- readRDS(here("models/overly_complex_glm_wicket_model.rds")
#summary(overly_complex_glm_wicket_model)
modelsummary(overly_complex_glm_wicket_model)
```

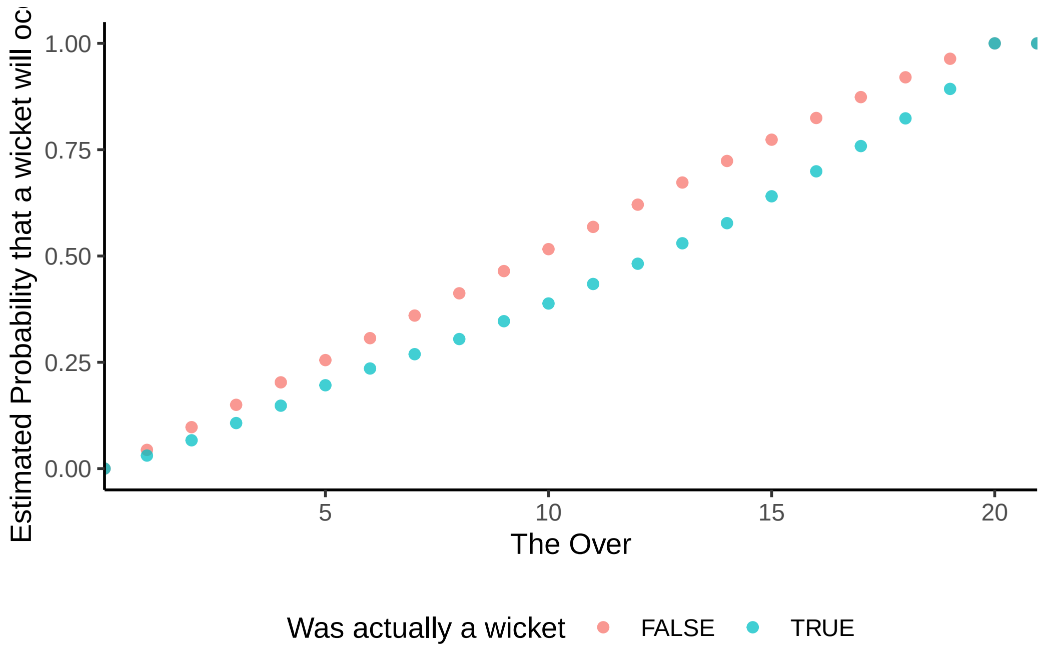
5 Simple Model Summary

```
overly_complex_glm_wicket_model <- complex_glm_wicket_model <- readRDS(here("models/overly_c

#summary(overly_complex_glm_wicket_model)
modelsummary(overly_complex_glm_wicket_model)
```

```
simple_glm_wicket_model_predictions <-
  predictions(simple_glm_wicket_model) |>
  as_tibble()

simple_glm_wicket_model_predictions |>
  mutate(wicket = factor(wicket)) |>
  ggplot(aes(x = over, y = estimate, color = wicket)) +
  stat_ecdf(geom = "point", alpha = 0.75) +
  labs(
    x = "The Over",
    y = "Estimated Probability that a wicket will occur",
    color = "Was actually a wicket"
  ) +
  theme_classic() +
  theme(legend.position = "bottom")
```

	(1)
(Intercept)	−4.316
	(0.192)
over	0.005
	(0.004)
prev_over_wickets	1.853
	(0.031)
batting_styleRight hand Bat	0.044
	(0.045)
bowling_styleLeft arm Fast medium	0.099
	(0.207)
bowling_styleLeft arm Medium	0.255
	(0.212)
bowling_styleLeft arm Medium fast	0.205
	(0.206)
bowling_styleLeft arm Wrist spin	0.158
	(0.235)
bowling_styleLegbreak	0.257
	(0.212)
bowling_styleLegbreak Googly	0.207
	(0.196)
bowling_styleRight arm Fast	0.190
	(0.192)
bowling_styleRight arm Fast medium	0.211
	(0.197)
bowling_styleRight arm Medium	0.109
	(0.193)
bowling_styleRight arm Medium fast	0.165
	(0.204)
bowling_styleRight arm Offbreak	0.104
	(0.200)
bowling_styleSlow Left arm Orthodox	0.011
	(0.202)
Num.Obs.	51 587
AIC	16 605.8
BIC	16 747.4
Log.Lik.	−8286.914
RMSE	0.21

	(1)
(Intercept)	−4.316
	(0.192)
over	0.005
	(0.004)
prev_over_wickets	1.853
	(0.031)
batting_styleRight hand Bat	0.044
	(0.045)
bowling_styleLeft arm Fast medium	0.099
	(0.207)
bowling_styleLeft arm Medium	0.255
	(0.212)
bowling_styleLeft arm Medium fast	0.205
	(0.206)
bowling_styleLeft arm Wrist spin	0.158
	(0.235)
bowling_styleLegbreak	0.257
	(0.212)
bowling_styleLegbreak Googly	0.207
	(0.196)
bowling_styleRight arm Fast	0.190
	(0.192)
bowling_styleRight arm Fast medium	0.211
	(0.197)
bowling_styleRight arm Medium	0.109
	(0.193)
bowling_styleRight arm Medium fast	0.165
	(0.204)
bowling_styleRight arm Offbreak	0.104
	(0.200)
bowling_styleSlow Left arm Orthodox	0.011
	(0.202)
Num.Obs.	51 587
AIC	16 605.8
BIC	16 747.4
Log.Lik.	−8286.914
RMSE	0.21

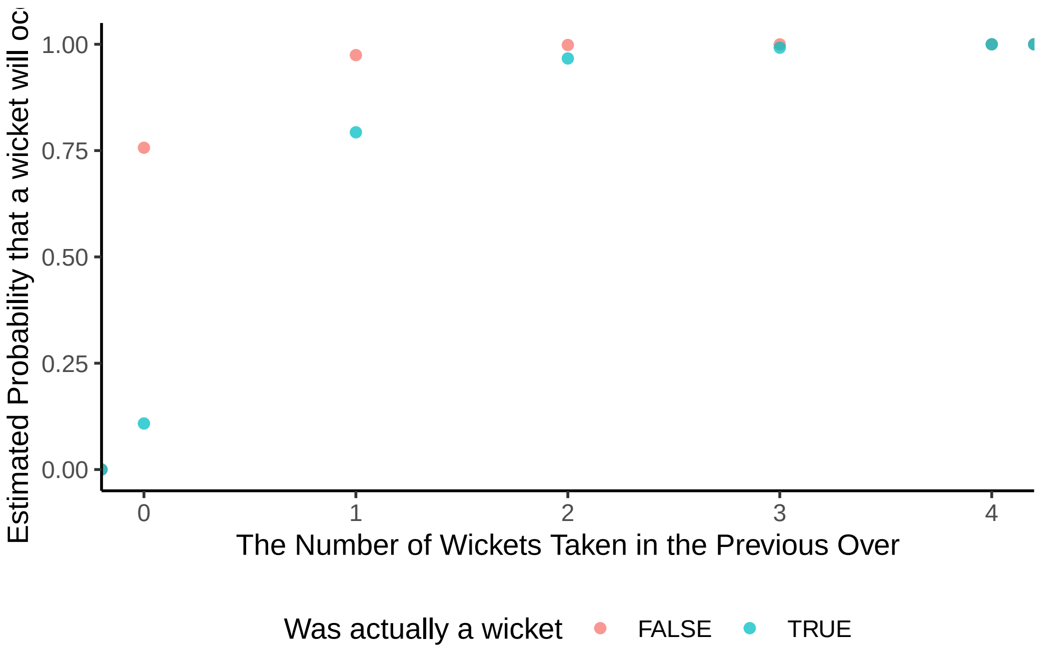


```
test_data_simple <- test_data
predictions <- predict(simple_glm_wicket_model, newdata = test_data_simple, type = "response'

test_data_simple$predicted_wicket_prob <- predictions
test_data_simple <- test_data_simple %>%
  mutate(predicted_wicket = predicted_wicket_prob >= 0.5) %>%
  mutate(correct_prediction = predicted_wicket == wicket)

summary_results <- test_data_simple %>% group_by(wicket) %>%
  summarise(
  correct = sum(correct_prediction),
  incorrect = sum(!correct_prediction)
)

summary_results
```

```
# A tibble: 2 x 3
  wicket correct incorrect
  <lgl>    <int>     <int>
1 FALSE    12253         0
2 TRUE         0       644
```

5.1 Complex Model Summary

```
complex_glm_wicket_model_predictions <-
  predictions(complex_glm_wicket_model) |>
  as_tibble()

complex_glm_wicket_model_predictions |>
  mutate(wicket = factor(wicket)) |>
  ggplot(aes(x = prev_over_wickets, y = estimate, color = wicket)) +
  stat_ecdf(geom = "point", alpha = 0.75) +
  labs(
    x = "The Number of Wickets Taken in the Previous Over",
    y = "Estimated Probability that a wicket will occur",
    color = "Was actually a wicket"
  ) +
  theme_classic() +
  theme(legend.position = "bottom")
```



```
test_data_complex <- test_data
predictions <- predict(complex_glm_wicket_model, newdata = test_data_complex, type = "respons

test_data_simple$predicted_wicket_prob <- predictions
test_data_simple <- test_data_simple %>%
  mutate(predicted_wicket = predicted_wicket_prob >= 0.5) %>%
  mutate(correct_prediction = predicted_wicket == wicket)

summary_results <- test_data_simple %>% group_by(wicket) %>%
  summarise(
  correct = sum(correct_prediction),
  incorrect = sum(!correct_prediction)
)

summary_results
```

```
# A tibble: 2 x 3
  wicket correct incorrect
  <lgl>    <int>     <int>
1 FALSE    12229        24
2 TRUE        17       627
```

6 Discussion

6.1 First discussion point

If my paper were 10 pages, then should be be at least 2.5 pages. The discussion is a chance to show off what you know and what you learnt from all this.

6.2 Second discussion point

Please don't use these as sub-heading labels - change them to be what your point actually is.

6.3 Third discussion point

6.4 Weaknesses and next steps

Weaknesses and next steps should also be included.

Appendix

A Data Sheet for The Raw Data

A.1 Questions

A.1.1 Who put the dataset together?

The raw dataset was acquired from the (**cricketdata?**) R package. This dataset had many contributors which can be seen through this link, but the main developer is Rob Hyndman. The dataset scraps the html from (**espncricinfo?**) and (**Cricsheet?**). These are two reputable organizations when it comes to cricket game data. (**Cricsheet?**) is moreso responsible for play by play cricket data, like giving the data for what happens for every ball in a game, whereas (**espncricinfo?**) is more responsible for general player career information and statistics.

A.1.2 Who paid for the dataset to be created?

The (**cricketdata?**) R package is completely open source and free under the GPL-3 license. The source code can be seen through this link. In terms of ESPN Cricinfo, this is a site and organization that makes money talking about cricket and airing games. So, it is in their best interest to give basic statistics for free on their website to entice people to use their services to watch and interact with content related to cricket. Cricsheet is operated by Stephen Rushe. He has written code to extract play-by-play cricket match data, which most likely means he is web scraping. There is not much other information about the code that extracts the data and what sites he uses.

A.1.3 How complete is the dataset?

The play by play data is very complete. For each ball in a game, it includes the bowler, batter, the batter on the opposite wicket, the venue, teams, the runs given, and more. The important data that seems to be missing is data related to the position of the fielders and the distance and final location of the ball once and if it hits the bat. In terms of player info, it is also very complete. It includes information about their batting style, blowing style if applicable, and their role on the team. The only issue here is that a player's role on the team can change depending on the team they are one. For instance, a player might be an opening batsman when playing in test cricket but could be a midover batsman when playing in T20. This dataset does not account for this discrepancy.

A.1.4 Which variables are present, and, equally, not present, for particular observations?

For play-by-play data the bowler, battsman, and teams involved in the play are present and so is general game information. Furthermore, the final outcome of the play, such as the number of runs, or if a wicket occurred are also given. Information about ball speed, distance, location, and fielding is not given. For player information, information about their country of citizenship, country of birth, batting style, bowling style, and batting order, and player role on a team are given. Potential missing observations with the player information are listed in Section A.1.3.

A.2 Data Dictionary For Raw Data

For this report, there are only two datasets used, the IPL (Indian Premier League) play-by-play data and player metadata. The variables for these data sets will be listed here with examples.

A.2.1 Play-by-play IPL Data

Match id (Integer)
Example: 335982
A randomly generated ID uniquely identifies each cricket match in the dataset.

Season (string)
Example: 2007/8 or 2024
Represents the IPL tournament year of the match. For instance "2007/8" represents the 2007 to 2008 IPL season and 2024 represents the tournament that was played only in 2024.

Season (date)
Example: 2024-04-15
Represents the day the match was played. It is in a year-month-day format.

Venue (string)
Example: M Chinnaswamy Stadium
The location the match is played in.

Innings (Integer)
Example: 1
The innings the current ball is played in. Should only be either 1 or two.

Over (Integer)
Example: 7
The over the current ball is played in. This should be from 1 to 20 in T20 Cricket.

Ball (Integer)
Example: 3

The current ball number in the over. This typically should be from 1 to 6, but could be more if no balls or wides are given.

Batting Team (String)
Example: Kings XI Punjab
The team that is currently batting.

Bowling Team (String)
Example: Mumbai Indians
The team that is currently bowling.

Striker (String)
Example: AJ Finch
The player that is currently on the wicket the bowler is bowling to.

Non-Striker (String)
Example: A Ashish Reddy
The player that is batting but is on the opposition wicket.

Bowler (String)
Example: SMSM Senanayake
The player that is bowling.

Runs off of the bat (Integer)
Example: 3
The number of the runs the team scored from this ball directly off of the strikers bat.

Extra Runs (Integer)
Example: 1
The number of runs that the team scored that was not because of the striker's bat. This could be because of wides.

Extra Ball (Boolean)
Example: False
Was this ball an extra ball? For instance, this ball could be extra due to a wide.

Balls Remaining (Integer)
Example: 79
The number of balls remaining in the game. This is from 0 to 120.

Runs Scored (Integer)
Example: 101
The number of runs already scored by the batting team.

Wicket (Boolean)
Example: True
Did a wicket occur in this ball.
Wickets Lost (Integer)

Example: 7
The number of wickets that have fallen. From 0 to 10.

First Innings Total (Integer)
Example: 165
The total number of runs earned in the first innings.

Second Innings Total (Integer)
Example: 168
The total number of runs earned in the second innings.

Target (Integer)
Example: 207
The par course for the given venue's pitch on that match's day.

Wides (Integer)
Example: 4
The number of wides bowled in this over.

No Balls (Integer)
Example: 2
The number of no balls bowled in this over.

Byes (Integer)
Example: 4
Runs scored by byes in this ball. This ranges from 0 to 4.

Leg Byes (Integer)
Example: 2
Runs scored by leg byes in this ball. This ranges from 0-4.

Wicket Type (String)
Example: Bowled
If a wicket occurred, this column will describe how it happened. It includes bowled, stumped, and caught.

Player Dismissed (String)
Example: A Ashish Reddy
If a wicket occurred, this column will have the name of the player that was dismissed.

A.2.2 Player Metadata

Cricinfo Player id (Integer)
Example: 1175501
A randomly generated ID uniquely identifies each cricket player from Cricinfo data.

Cricsheet Player id (Alphanumeric)
Example: 21d38d47
A randomly generated ID uniquely identifies each cricket player from Cricinfo data.

Unique Name (String)
Example: Alkandari Abdulrahman
The unique name of the player that is stored in the (**cricketdata?**) package. All other tables will only use the unique name of a player and not the full name of the player.

Full Name (String)
Example: AGHM Alkandari Abdulrahman
The full name of a player.

Country (String)
Example: Zimbabwe
The country of citizenship for this player.

Date of Birth (Date)
Example: 1971-09-21
The day the player was born in a year-month-day format.

Birthplace (String)
Example: Nangrahar, Afghanistan
The city and sometimes country a player was born in.

Batting Style (String)
Example: Left hand Bat
The batting style of a player. This is either left hand or right hand.

Bowling Style (String)
Example: Slow Left arm Orthodox
The bowling style of a player. This only exists if a player is a bowler.

Player Role (String)
Example: Allrounder
The role of a player on the team.

B Additional data details

```
bowling_batting_role_matchup_boundaries <- cleaned_data %>%
  group_by(bowling_style, batter_playing_role) %>%
  summarise(
    num_wickets = sum(wicket == TRUE),
```

```r
    num_balls = n(),
  ) %>% arrange(bowling_style, batter_playing_role)
```

```
`summarise()` has grouped output by 'bowling_style'. You can override using the
`.groups` argument.
```

```r
bowling_batting_role_matchup_boundaries
```

```
# A tibble: 117 x 4
# Groups:   bowling_style [13]
   bowling_style        batter_playing_role num_wickets num_balls
   <chr>                <chr>                     <int>     <int>
 1 Left arm Fast        Allrounder                   10       115
 2 Left arm Fast        Batter                        2        95
 3 Left arm Fast        Batting Allrounder            2        79
 4 Left arm Fast        Bowler                        4        37
 5 Left arm Fast        Bowling Allrounder            2        18
 6 Left arm Fast        Middle order Batter           4        89
 7 Left arm Fast        Opening Batter                4       174
 8 Left arm Fast        Top order Batter              9       137
 9 Left arm Fast        Wicketkeeper Batter          12       190
10 Left arm Fast medium Allrounder                   25       503
# i 107 more rows
```

C Model details

C.1 Posterior predictive check

D References

What to cite: - cricketdata - ESPNCricinfo - Cricsheet - All tidyverse packages used