F1_analysis

2025-12-07

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
#Shiv
#setwd("/Users/shivpatel/Desktop/Code")

# --------------------------
# Libraries
# --------------------------
library(httr)
library(jsonlite)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(purrr)
```

```
##
## Attaching package: 'purrr'
```

```
## The following object is masked from 'package:jsonlite':
##
##     flatten
```

```
library(tidyr)
library(openmeteo)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WB
a
```

```
library(rpart)
library(partykit)
```

```
## Loading required package: grid
```

```
## Loading required package: libcoin
```

```
## Loading required package: mvtnorm
```

```
library(party)
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
##
## Attaching package: 'party'
```

```
## The following objects are masked from 'package:partykit':
##
##     cforest, ctree, ctree_control, edge_simple, mob, mob_control,
##     node_barplot, node_bivplot, node_boxplot, node_inner, node_surv,
##     node_terminal, varimp
```

```
## The following object is masked from 'package:dplyr':
##
##     where
```

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
# ------------------------
# Function to import a given years data frame and then join it onto master
# ------------------------
import_clean <- function(frames,year){
  # ------------------------
  # Get all Race sessions for the year
  # ------------------------
  sessions_url <- paste0("https://api.openf1.org/v1/sessions?year=",year,"&session_name=
Race")

  sessions_raw <- fromJSON(content(GET(sessions_url), "text", encoding = "UTF-8"))
  sessions <- sessions_raw

  # ------------------------
  # Position and Driver Data
  # ------------------------
  results <- map_df(sessions_raw$session_key,
                    function(ses){
                      Sys.sleep(0.5)
                      url <- paste0('https://api.openf1.org/v1/drivers?session_key=',se
s)
                      df <- fromJSON(content(GET(url), "text", encoding = "UTF-8"))
                      df$session_key <- ses
                      return(df)})

  drivers <- results


  results <- map_df(sessions_raw$session_key,
                    function(ses){
                      Sys.sleep(0.5)
                      url <- paste0("https://api.openf1.org/v1/position?session_key=",se
s)
                      df <- fromJSON(content(GET(url), "text", encoding = "UTF-8"))
                      df$session_key <- ses
                      return(df)})

  position <- results


  driver_pos <- left_join(position, drivers,  by = join_by(session_key==session_key, dri
ver_number==driver_number))
  driver_pos <- select(driver_pos,"date","session_key","driver_number","position","full_
name","team_name")

  driver_pos <- driver_pos %>%
    group_by(session_key,driver_number) %>%
    filter(date == max(date)) %>%
    pivot_wider(
      id_cols = session_key,
      names_from = position,
      values_from = c(driver_number,full_name,team_name),
```

```
      names_prefix = "pos_")


  # -------------------------
  # Weather
  # -------------------------
  results <- map_df(sessions_raw$session_key,
                    function(ses){
                      Sys.sleep(0.5)
                      url <- paste0('https://api.openf1.org/v1/weather?session_key=',se
s)
                      df <- fromJSON(content(GET(url), "text", encoding = "UTF-8"))
                      df$session_key <- ses
                      return(df)})

  weather <- results


  weather <- weather %>%
    group_by(session_key) %>%
    summarize( wind_direction = mean(wind_direction, na.rm = TRUE),
               meeting_key = max(meeting_key, na.rm = TRUE),
               wind_speed = mean(wind_speed, na.rm = TRUE),
               #Rainfall is normally a binary denoteing if in a given minute there was r
ain. This new rainfall variable demotes the percentage of minutes in the race that had r
ain
               track_temperature = mean(track_temperature, na.rm = TRUE),
               rainfall = sum(rainfall, na.rm = TRUE)/length(rainfall),
               air_temperature = mean(air_temperature, na.rm = TRUE),
               humidity = mean(humidity, na.rm = TRUE),
               pressure = mean(pressure, na.rm = TRUE))

  # -------------------------
  # Get all qualifying sessions for the year Qualifying
  # -------------------------
  results <- map_df(sessions_raw$meeting_key,
                    function(meet){
                      Sys.sleep(0.5)
                      url <- paste0("https://api.openf1.org/v1/sessions?&session_name=Qu
alifying&meeting_key=",meet)
                      df <- fromJSON(content(GET(url), "text", encoding = "UTF-8"))
                      df$meeting_key <- meet
                      return(df)})

  quals <- results[c(1,2)]

  # -------------------------
  # Qual_results
  # -------------------------
  results <- map_df(quals$session_key,
                    function(ses){
                      Sys.sleep(0.5)
```

```
                    url <- paste0("https://api.openf1.org/v1/position?session_key=",se
s)
                    df <- fromJSON(content(GET(url), "text", encoding = "UTF-8"))
                    #df$meeting_key <- ses
                    return(df)})
  quals_results <- results

  quals_results <- left_join(quals_results, drivers,  by = join_by(meeting_key==meeting_
key, driver_number==driver_number)) %>%
    select("date","meeting_key","session_key.x","driver_number","position","full_nam
e","team_name")

  quals_results <- quals_results %>%
    group_by(session_key.x,driver_number) %>%
    filter(date == max(date))
  quals_results_2 <- quals_results
  quals_results <- pivot_wider(quals_results,
      id_cols = c(session_key.x,meeting_key),
      names_from = position,
      values_from = c(driver_number,full_name,team_name),
      names_prefix = "qual_pos_")


  # -------------------------
  # 4. Merge into final DF
  # -------------------------

  full_df <- sessions %>%
    left_join(driver_pos,  by = "session_key") %>%
    left_join(weather,  by = "session_key")%>%
    left_join(quals_results, by = c("meeting_key.x"="meeting_key"))

  full_df <- select(full_df,meeting_key = "meeting_key.x",
                    "session_key.x",
                    "location",
                    "date_start",
                    "date_end",
                    "country_key",
                    "country_name",
                    "circuit_key",
                    "circuit_short_name",
                    "year",
                    "driver_number_pos_20",
                    "driver_number_pos_19",
                    "driver_number_pos_18",
                    "driver_number_pos_17",
                    "driver_number_pos_16",
                    "driver_number_pos_15",
                    "driver_number_pos_14",
                    "driver_number_pos_13",
                    "driver_number_pos_12",
                    "driver_number_pos_11",
```

```
                "driver_number_pos_10",
                "driver_number_pos_9",
                "driver_number_pos_8",
                "driver_number_pos_7",
                "driver_number_pos_6",
                "driver_number_pos_5",
                "driver_number_pos_4",
                "driver_number_pos_3",
                "driver_number_pos_2",
                "driver_number_pos_1",
                "full_name_pos_20",
                "full_name_pos_19",
                "full_name_pos_18",
                "full_name_pos_17",
                "full_name_pos_16",
                "full_name_pos_15",
                "full_name_pos_14",
                "full_name_pos_13",
                "full_name_pos_12",
                "full_name_pos_11",
                "full_name_pos_10",
                "full_name_pos_9",
                "full_name_pos_8",
                "full_name_pos_7",
                "full_name_pos_6",
                "full_name_pos_5",
                "full_name_pos_4",
                "full_name_pos_3",
                "full_name_pos_2",
                "full_name_pos_1",
                "team_name_pos_20",
                "team_name_pos_19",
                "team_name_pos_18",
                "team_name_pos_17",
                "team_name_pos_16",
                "team_name_pos_15",
                "team_name_pos_14",
                "team_name_pos_13",
                "team_name_pos_12",
                "team_name_pos_11",
                "team_name_pos_10",
                "team_name_pos_9" ,
                "team_name_pos_8",
                "team_name_pos_7",
                "team_name_pos_6" ,
                "team_name_pos_5",
                "team_name_pos_4",
                "team_name_pos_3",
                "team_name_pos_2",
                "team_name_pos_1",
                "driver_number_qual_pos_20",
                "driver_number_qual_pos_19",
```

```
                "driver_number_qual_pos_18",
                "driver_number_qual_pos_17",
                "driver_number_qual_pos_16",
                "driver_number_qual_pos_15",
                "driver_number_qual_pos_14",
                "driver_number_qual_pos_13",
                "driver_number_qual_pos_12",
                "driver_number_qual_pos_11",
                "driver_number_qual_pos_10",
                "driver_number_qual_pos_9",
                "driver_number_qual_pos_8",
                "driver_number_qual_pos_7",
                "driver_number_qual_pos_6",
                "driver_number_qual_pos_5",
                "driver_number_qual_pos_4",
                "driver_number_qual_pos_3",
                "driver_number_qual_pos_2",
                "driver_number_qual_pos_1",
                "full_name_qual_pos_20",
                "full_name_qual_pos_19",
                "full_name_qual_pos_18",
                "full_name_qual_pos_17",
                "full_name_qual_pos_16",
                "full_name_qual_pos_15",
                "full_name_qual_pos_14",
                "full_name_qual_pos_13",
                "full_name_qual_pos_12",
                "full_name_qual_pos_11",
                "full_name_qual_pos_10",
                "full_name_qual_pos_9",
                "full_name_qual_pos_8",
                "full_name_qual_pos_7",
                "full_name_qual_pos_6",
                "full_name_qual_pos_5",
                "full_name_qual_pos_4",
                "full_name_qual_pos_3",
                "full_name_qual_pos_2",
                "full_name_qual_pos_1",
                "team_name_qual_pos_20",
                "team_name_qual_pos_19",
                "team_name_qual_pos_18",
                "team_name_qual_pos_17",
                "team_name_qual_pos_16",
                "team_name_qual_pos_15",
                "team_name_qual_pos_14",
                "team_name_qual_pos_13",
                "team_name_qual_pos_12",
                "team_name_qual_pos_11",
                "team_name_qual_pos_10",
                "team_name_qual_pos_9" ,
                "team_name_qual_pos_8",
                "team_name_qual_pos_7",
```

```
                    "team_name_qual_pos_6" ,
                    "team_name_qual_pos_5",
                    "team_name_qual_pos_4",
                    "team_name_qual_pos_3",
                    "team_name_qual_pos_2",
                    "team_name_qual_pos_1",
                    "wind_direction",
                    "wind_speed",
                    "rainfall",
                    "track_temperature",
                    "air_temperature",
                    "humidity",
                    "pressure")

  master_df_tall <- left_join(drivers,position,join_by(session_key,meeting_key,driver_nu
mber)) %>%
    left_join(sessions,join_by(session_key,meeting_key)) %>%
    left_join(weather,join_by(session_key,meeting_key)) %>%
    left_join(quals_results_2,join_by(meeting_key,driver_number))

  return(list(rbind(as.data.frame(frames[1]),full_df),rbind(as.data.frame(frames[2]),mas
ter_df_tall)))}

# -------------------------
# Import our needed years
# -------------------------
## Import if not saved locally
    frames <- list(data.frame(),data.frame())
    frames <- import_clean(frames,2023)
    frames <- import_clean(frames,2024)
    frames <- import_clean(frames,2025)

    master_df <- as.data.frame(frames[1])
    master_df_tall <- as.data.frame(frames[2])
  #
  # saveRDS(master_df, "master_hist_table.rds")
  # saveRDS(master_df_tall, "master_hist_table_tall.rds")

#Import locally stored file (saves API calls)
#master_df <- readRDS("master_hist_table.rds")
#master_df_tall <- readRDS("master_hist_table_tall.rds")
#Shiv

# -------------------------
# Clustering locations
# -------------------------
#Temperature difference may be more useful than individual temps
master_df$temp_diff <- master_df$track_temperature - master_df$air_temperature

#Aggregate track temps
track_features_agg <- master_df %>%
  group_by(location) %>%
```

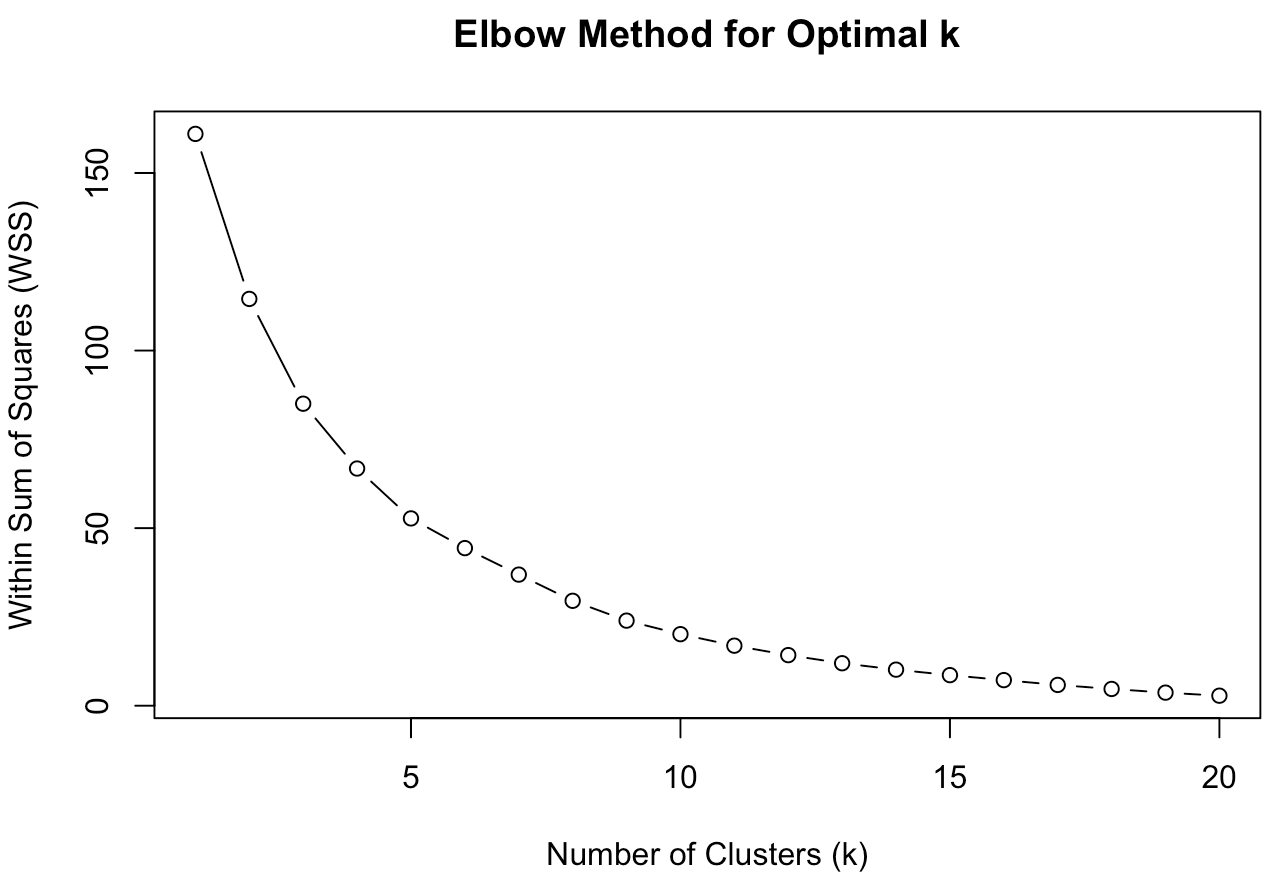
```
  summarise(
    Avg_Wind_Speed = mean(wind_speed, na.rm = TRUE),
    Avg_Rainfall_Perc = mean(rainfall, na.rm = TRUE),
    Avg_Track_Temp = mean(track_temperature, na.rm = TRUE),
    Avg_Air_Temp = mean(air_temperature, na.rm = TRUE),
    Avg_Humidity = mean(humidity, na.rm = TRUE),
    Avg_Pressure = mean(pressure, na.rm = TRUE),
    Avg_Temp_Diff = mean(temp_diff, na.rm = TRUE),
    .groups = 'drop'
  )

clustering_data <- track_features_agg %>%
  select(starts_with("Avg_")) %>%
  na.omit()

scaled_data <- scale(clustering_data)


wss <- numeric(20)
for (k in 1:20) {
  kmeans_result <- kmeans(scaled_data, centers = k, nstart = 25, iter.max = 30)
  wss[k] <- kmeans_result$tot.withinss
}
print("WSS Values for k=1 to 10:")
```

```
## [1] "WSS Values for k=1 to 10:"
```

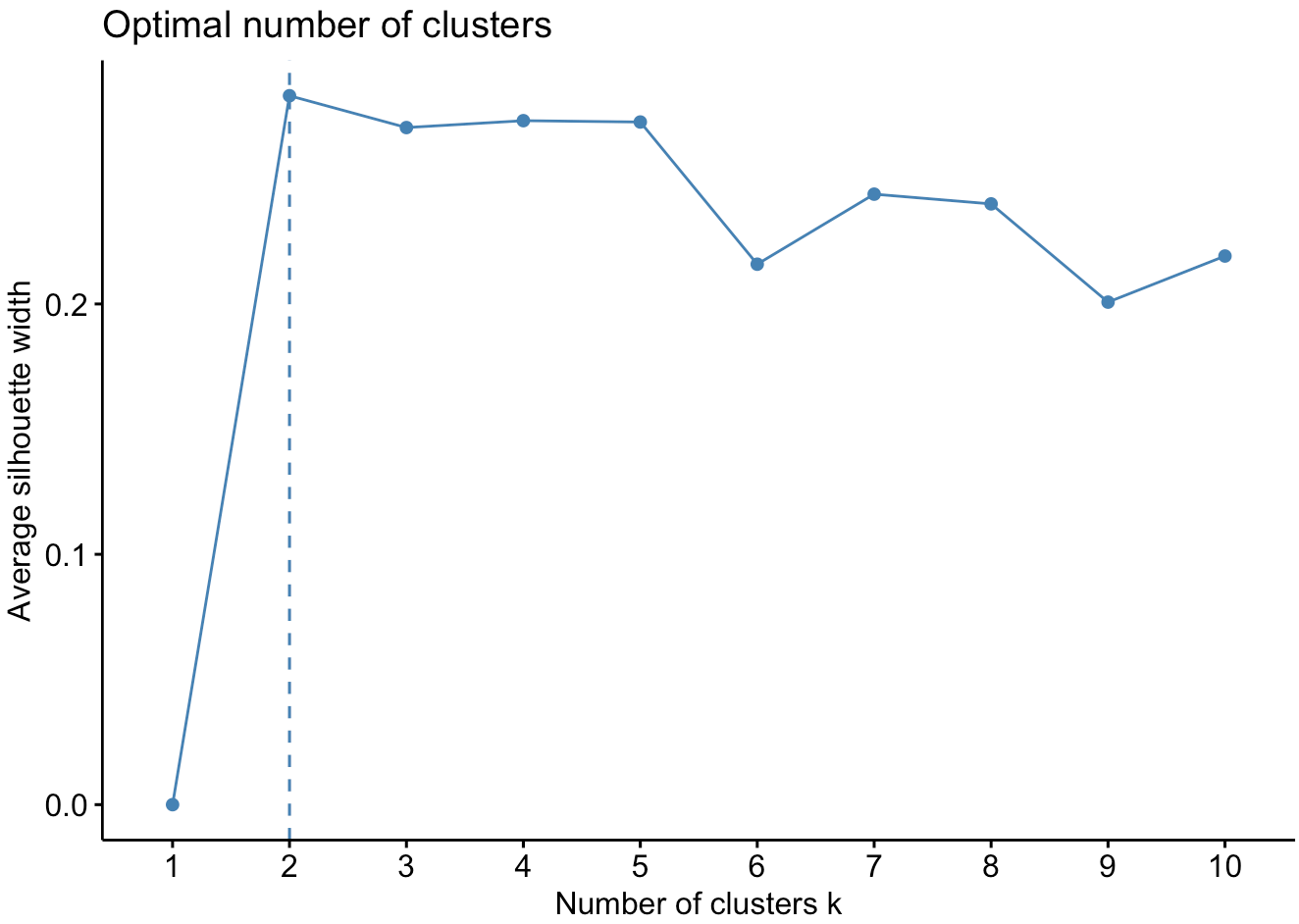
```
print(wss)
```

```
##  [1] 161.000000 114.542128  85.036998  66.785549  52.738398  44.378923
##  [7]  36.914603  29.538929  23.957175  20.145769  16.906088  14.248345
## [13]  11.962048  10.158705   8.612900   7.210594   5.852706   4.718458
## [19]   3.684299   2.822952
```

```
plot(1:20, wss, type="b", xlab="Number of Clusters (k)", ylab="Within Sum of Squares (WSS)", main="Elbow Method for Optimal k")
```



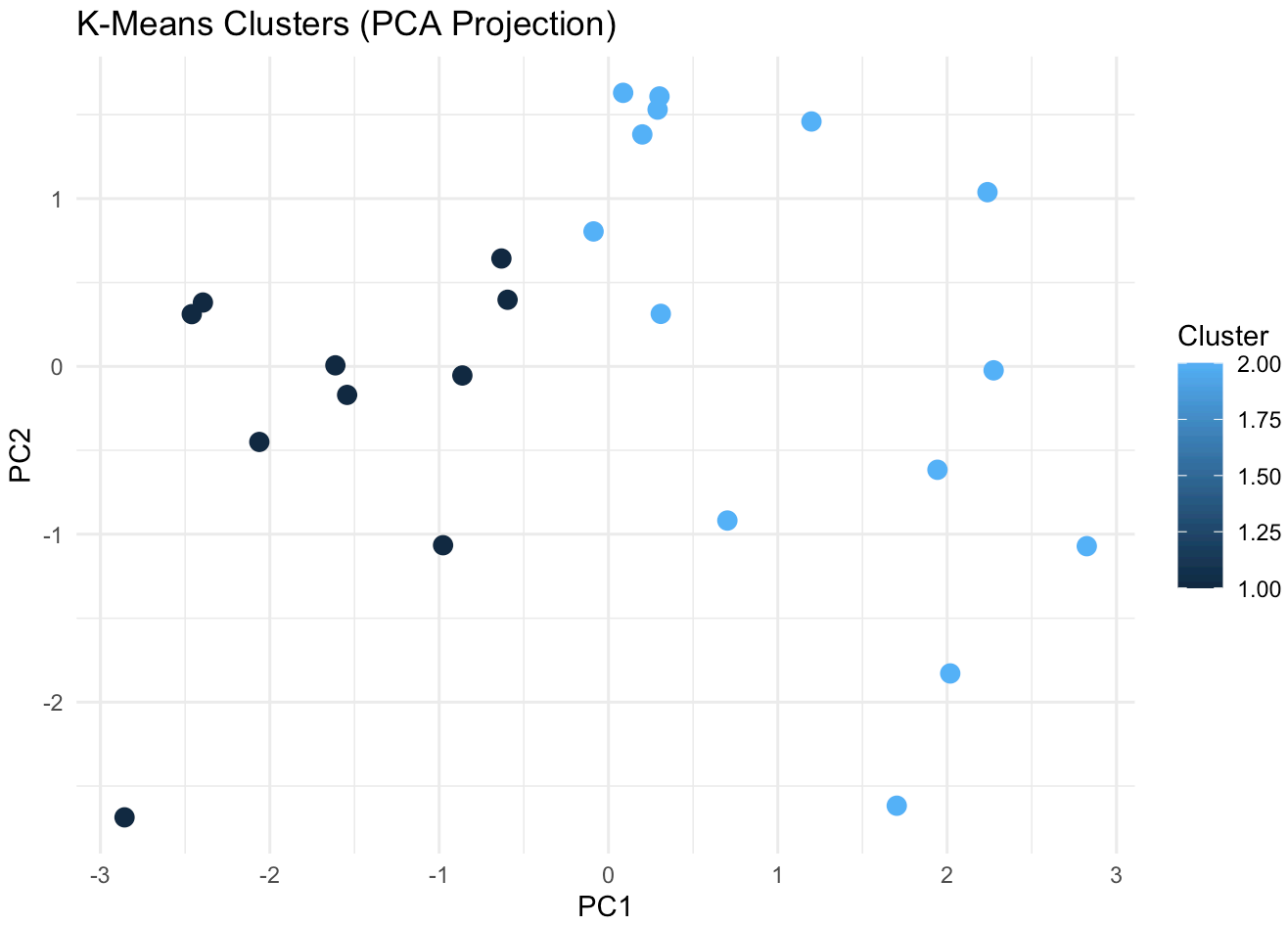

```
fviz_nbclust(scaled_data,FUNcluster = kmeans)
```




```
k_optimal <- 2

final_clusters <- kmeans(scaled_data, centers = k_optimal, nstart = 25, iter.max = 30)


cluster_assignments <- track_features_agg %>%
  filter(complete.cases(select(., starts_with("Avg_")))) %>%
  mutate(Track_Cluster = final_clusters$cluster) %>%
  select(location, Track_Cluster, everything())

print(cluster_assignments)
```

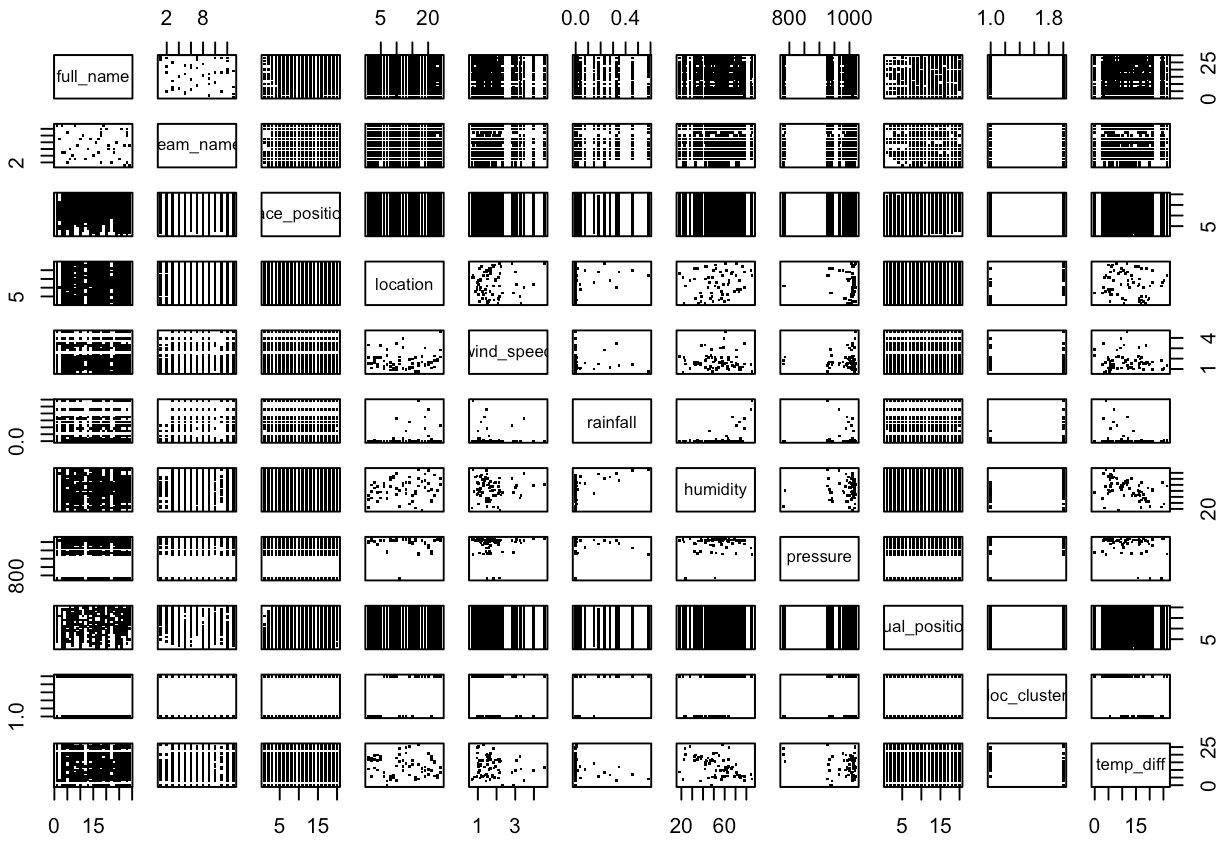
```
## # A tibble: 24 × 9
##    location   Track_Cluster Avg_Wind_Speed Avg_Rainfall_Perc Avg_Track_Temp
##    <chr>              <int>          <dbl>             <dbl>          <dbl>
##  1 Austin                 1           2.30           0                44.1
##  2 Baku                   1           1.38           0                37.2
##  3 Barcelona              1           1.88           0.00216          41.2
##  4 Budapest               1           1.59           0                41.8
##  5 Imola                  1           2.69           0                42.3
##  6 Jeddah                 2           1.37           0                33.7
##  7 Las Vegas              2           2.21           0                18.4
##  8 Lusail                 2           1.27           0                29.3
##  9 Marina Bay             2           1.16           0.0124           35.6
## 10 Melbourne              2           1.85           0.109            29.2
## # ℹ 14 more rows
## # ℹ 4 more variables: Avg_Air_Temp <dbl>, Avg_Humidity <dbl>,
## #   Avg_Pressure <dbl>, Avg_Temp_Diff <dbl>
```

```
pca <- prcomp(scaled_data)
pca_df <- data.frame(
  PC1 = pca$x[, 1],
  PC2 = pca$x[, 2],
  Cluster = cluster_assignments$Track_Cluster
)

ggplot(pca_df, aes(PC1, PC2, color = Cluster)) +
  geom_point(size = 3) +
  theme_minimal() +
  ggtitle("K-Means Clusters (PCA Projection)")
```


K-Means Clusters (PCA Projection)
PC2
PC1
1
0
-1
-2
-3
-2
-1
0
1
2
3
Cluster
2.00
1.75
1.50
1.25
1.00

```
master_df <- left_join(master_df,cluster_assignments,by="location")




# -------------------------
# Add in additional variables
# -------------------------
analysis_set <- left_join(master_df_tall,cluster_assignments,by="location")
analysis_set <- select(analysis_set,"driver_number",
                       full_name="full_name.x",
                       team_name = "team_name.x",
                       race_position = "position.x",
                       "location",
                       "year",
                       "wind_speed",
                       "track_temperature",
                       "rainfall",
                       "air_temperature",
                       "humidity",
                       "pressure",
                       qual_position = "position.y",
                       loc_cluster = Track_Cluster)
analysis_set$temp_diff <- analysis_set$track_temperature - analysis_set$air_temperature
analysis_set$driver_number <- as.factor(analysis_set$driver_number)
analysis_set$full_name <- as.factor(analysis_set$full_name)
analysis_set$team_name <- as.factor(analysis_set$team_name)
analysis_set$location <- as.factor(analysis_set$location)
analysis_set$year <- as.factor(analysis_set$year)
analysis_set$loc_cluster <- as.factor(analysis_set$loc_cluster)



# -------------------------
# Make Train and Test Sets
# -------------------------
set.seed(112224)
ind <- sample(1:nrow(analysis_set), size = 0.7*nrow(analysis_set), replace = FALSE)
test <- analysis_set[-ind,]
train <- analysis_set[ind,]



# -------------------------
# Linear Model
# -------------------------

# ##
#First look for inter-relationship patterns
```

```
# ##
pairs(~., data=train[-c(1,6,8,10)], pch=".")
```




```
# ##
# Model using qualifying results
# ##
lm_wqual <- lm(race_position ~ ., data = train[c("team_name","wind_speed","rainfall","hu
midity","pressure","qual_position","loc_cluster","temp_diff","race_position")])
#N:Obviously large amount of multicolinearity between name and team
summary(lm_wqual)
```

```
##
## Call:
## lm(formula = race_position ~ ., data = train[c("team_name", "wind_speed",
##     "rainfall", "humidity", "pressure", "qual_position", "loc_cluster",
##     "temp_diff", "race_position")])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5659  -2.4499  -0.2054   2.2924  14.4609
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                5.6973695  0.4846213  11.756  < 2e-16 ***
## team_nameAlphaTauri       -0.4602610  0.1620438  -2.840 0.004510 **
## team_nameAlpine           -0.3645462  0.1315489  -2.771 0.005590 **
## team_nameAston Martin     -0.7557428  0.1318753  -5.731 1.01e-08 ***
## team_nameFerrari          -2.4410510  0.1438583 -16.968  < 2e-16 ***
## team_nameHaas F1 Team     -0.2399908  0.1296182  -1.852 0.064107 .
## team_nameKick Sauber       0.3745841  0.1422188   2.634 0.008448 **
## team_nameMcLaren          -2.0402580  0.1438182 -14.186  < 2e-16 ***
## team_nameMercedes         -1.7280713  0.1396832 -12.371  < 2e-16 ***
## team_nameRacing Bulls      0.1275074  0.1717690   0.742 0.457902
## team_nameRB                0.1649315  0.1653202   0.998 0.318460
## team_nameRed Bull Racing  -2.3387322  0.1403298 -16.666  < 2e-16 ***
## team_nameWilliams         -0.1639976  0.1315355  -1.247 0.212485
## wind_speed                 0.0113492  0.0301699   0.376 0.706789
## rainfall                   0.2097703  0.2713171   0.773 0.439438
## humidity                  -0.0023706  0.0018234  -1.300 0.193573
## pressure                   0.0018329  0.0004721   3.882 0.000104 ***
## qual_position              0.4557103  0.0052902  86.142  < 2e-16 ***
## loc_cluster2              -0.2660195  0.0592456  -4.490 7.15e-06 ***
## temp_diff                 -0.0186156  0.0045725  -4.071 4.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.538 on 23574 degrees of freedom
## Multiple R-squared:  0.4347, Adjusted R-squared:  0.4342
## F-statistic:   954 on 19 and 23574 DF,  p-value: < 2.2e-16
```

```
#Step
lm_race_step_wqual <- step(lm_wqual, scope=list(lower=lm(train$race_position~0), upper=l
m_wqual), direction = "forward")
```

```
## Start:  AIC=59639.09
## race_position ~ team_name + wind_speed + rainfall + humidity +
##     pressure + qual_position + loc_cluster + temp_diff
```

```
summary(lm_race_step_wqual)
```

```
##
## Call:
## lm(formula = race_position ~ team_name + wind_speed + rainfall +
##     humidity + pressure + qual_position + loc_cluster + temp_diff,
##     data = train[c("team_name", "wind_speed", "rainfall", "humidity",
##         "pressure", "qual_position", "loc_cluster", "temp_diff",
##         "race_position")])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5659  -2.4499  -0.2054   2.2924  14.4609
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                5.6973695  0.4846213  11.756  < 2e-16 ***
## team_nameAlphaTauri       -0.4602610  0.1620438  -2.840 0.004510 **
## team_nameAlpine           -0.3645462  0.1315489  -2.771 0.005590 **
## team_nameAston Martin     -0.7557428  0.1318753  -5.731 1.01e-08 ***
## team_nameFerrari          -2.4410510  0.1438583 -16.968  < 2e-16 ***
## team_nameHaas F1 Team     -0.2399908  0.1296182  -1.852 0.064107 .
## team_nameKick Sauber       0.3745841  0.1422188   2.634 0.008448 **
## team_nameMcLaren          -2.0402580  0.1438182 -14.186  < 2e-16 ***
## team_nameMercedes         -1.7280713  0.1396832 -12.371  < 2e-16 ***
## team_nameRacing Bulls      0.1275074  0.1717690   0.742 0.457902
## team_nameRB                0.1649315  0.1653202   0.998 0.318460
## team_nameRed Bull Racing  -2.3387322  0.1403298 -16.666  < 2e-16 ***
## team_nameWilliams         -0.1639976  0.1315355  -1.247 0.212485
## wind_speed                 0.0113492  0.0301699   0.376 0.706789
## rainfall                   0.2097703  0.2713171   0.773 0.439438
## humidity                  -0.0023706  0.0018234  -1.300 0.193573
## pressure                   0.0018329  0.0004721   3.882 0.000104 ***
## qual_position              0.4557103  0.0052902  86.142  < 2e-16 ***
## loc_cluster2              -0.2660195  0.0592456  -4.490 7.15e-06 ***
## temp_diff                 -0.0186156  0.0045725  -4.071 4.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.538 on 23574 degrees of freedom
## Multiple R-squared:  0.4347, Adjusted R-squared:  0.4342
## F-statistic:   954 on 19 and 23574 DF,  p-value: < 2.2e-16
```

```
#without qualifying results
lm_race <- lm(race_position ~ ., data = train[c("team_name","wind_speed","rainfall","hum
idity","pressure","loc_cluster","temp_diff","race_position")])
summary(lm_race)
```

```
##
## Call:
## lm(formula = race_position ~ ., data = train[c("team_name", "wind_speed",
##     "rainfall", "humidity", "pressure", "loc_cluster", "temp_diff",
##     "race_position")])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.3283  -3.0199  -0.0893   2.8667  12.5390
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              12.5829315  0.5480612  22.959  < 2e-16 ***
## team_nameAlphaTauri      -0.3947155  0.1857990  -2.124 0.033645 *
## team_nameAlpine          -1.0708273  0.1505420  -7.113 1.17e-12 ***
## team_nameAston Martin    -2.0193805  0.1502711 -13.438  < 2e-16 ***
## team_nameFerrari         -6.1347353  0.1574517 -38.963  < 2e-16 ***
## team_nameHaas F1 Team    -0.5266671  0.1485725  -3.545 0.000394 ***
## team_nameKick Sauber      0.7611597  0.1629883   4.670 3.03e-06 ***
## team_nameMcLaren         -5.6996766  0.1575452 -36.178  < 2e-16 ***
## team_nameMercedes        -5.3218814  0.1528517 -34.817  < 2e-16 ***
## team_nameRacing Bulls    -1.6070348  0.1955941  -8.216  < 2e-16 ***
## team_nameRB              -0.7013402  0.1892067  -3.707 0.000210 ***
## team_nameRed Bull Racing -5.0293234  0.1568673 -32.061  < 2e-16 ***
## team_nameWilliams        -0.4610096  0.1507681  -3.058 0.002233 **
## wind_speed                0.0098093  0.0345931   0.284 0.776748
## rainfall                  0.6914275  0.3110289   2.223 0.026223 *
## humidity                 -0.0007400  0.0020906  -0.354 0.723365
## pressure                  0.0016104  0.0005413   2.975 0.002934 **
## loc_cluster2             -0.4113674  0.0679040  -6.058 1.40e-09 ***
## temp_diff                -0.0209022  0.0052428  -3.987 6.72e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.056 on 23575 degrees of freedom
## Multiple R-squared:  0.2567, Adjusted R-squared:  0.2562
## F-statistic: 452.4 on 18 and 23575 DF,  p-value: < 2.2e-16
```

```
lm_race_step <- step(lm_race, scope=list(lower=lm(train$race_position~0), upper=lm_rac
e), direction = "forward")
```

```
## Start:  AIC=66093.87
## race_position ~ team_name + wind_speed + rainfall + humidity +
##     pressure + loc_cluster + temp_diff
```

```
summary(lm_race_step)
```

```
##
## Call:
## lm(formula = race_position ~ team_name + wind_speed + rainfall +
##     humidity + pressure + loc_cluster + temp_diff, data = train[c("team_name",
##     "wind_speed", "rainfall", "humidity", "pressure", "loc_cluster",
##     "temp_diff", "race_position")])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.3283  -3.0199  -0.0893   2.8667  12.5390
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               12.5829315  0.5480612  22.959  < 2e-16 ***
## team_nameAlphaTauri       -0.3947155  0.1857990  -2.124 0.033645 *
## team_nameAlpine           -1.0708273  0.1505420  -7.113 1.17e-12 ***
## team_nameAston Martin     -2.0193805  0.1502711 -13.438  < 2e-16 ***
## team_nameFerrari          -6.1347353  0.1574517 -38.963  < 2e-16 ***
## team_nameHaas F1 Team     -0.5266671  0.1485725  -3.545 0.000394 ***
## team_nameKick Sauber       0.7611597  0.1629883   4.670 3.03e-06 ***
## team_nameMcLaren          -5.6996766  0.1575452 -36.178  < 2e-16 ***
## team_nameMercedes         -5.3218814  0.1528517 -34.817  < 2e-16 ***
## team_nameRacing Bulls     -1.6070348  0.1955941  -8.216  < 2e-16 ***
## team_nameRB               -0.7013402  0.1892067  -3.707 0.000210 ***
## team_nameRed Bull Racing  -5.0293234  0.1568673 -32.061  < 2e-16 ***
## team_nameWilliams         -0.4610096  0.1507681  -3.058 0.002233 **
## wind_speed                 0.0098093  0.0345931   0.284 0.776748
## rainfall                   0.6914275  0.3110289   2.223 0.026223 *
## humidity                  -0.0007400  0.0020906  -0.354 0.723365
## pressure                   0.0016104  0.0005413   2.975 0.002934 **
## loc_cluster2              -0.4113674  0.0679040  -6.058 1.40e-09 ***
## temp_diff                 -0.0209022  0.0052428  -3.987 6.72e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.056 on 23575 degrees of freedom
## Multiple R-squared:  0.2567, Adjusted R-squared:  0.2562
## F-statistic: 452.4 on 18 and 23575 DF,  p-value: < 2.2e-16
```

```
# Error
train$pred_wqual <- predict(lm_race_step_wqual,train[c("team_name","wind_speed","rainfal
l","humidity","pressure","qual_position","loc_cluster","temp_diff")])
train$pred <- predict(lm_race_step,train[c("team_name","wind_speed","rainfall","humidit
y","pressure","loc_cluster","temp_diff")])

test$pred_wqual <- predict(lm_race_step_wqual,test[c("team_name","wind_speed","rainfal
l","humidity","pressure","qual_position","loc_cluster","temp_diff")])
test$pred <- predict(lm_race_step,test[c("team_name","wind_speed","rainfall","humidit
y","pressure","loc_cluster","temp_diff")])

#Train
accuracy(train$pred_wqual,train$race_position)
```

```
##                    ME     RMSE      MAE       MPE    MAPE
## Test set 3.22807e-13 3.536062 2.803745 -21.15566 39.6356
```

```
accuracy(train$pred,train$race_position)
```

```
##                     ME    RMSE      MAE       MPE     MAPE
## Test set 1.003642e-12 4.05457 3.318377 -28.79826 49.99304
```

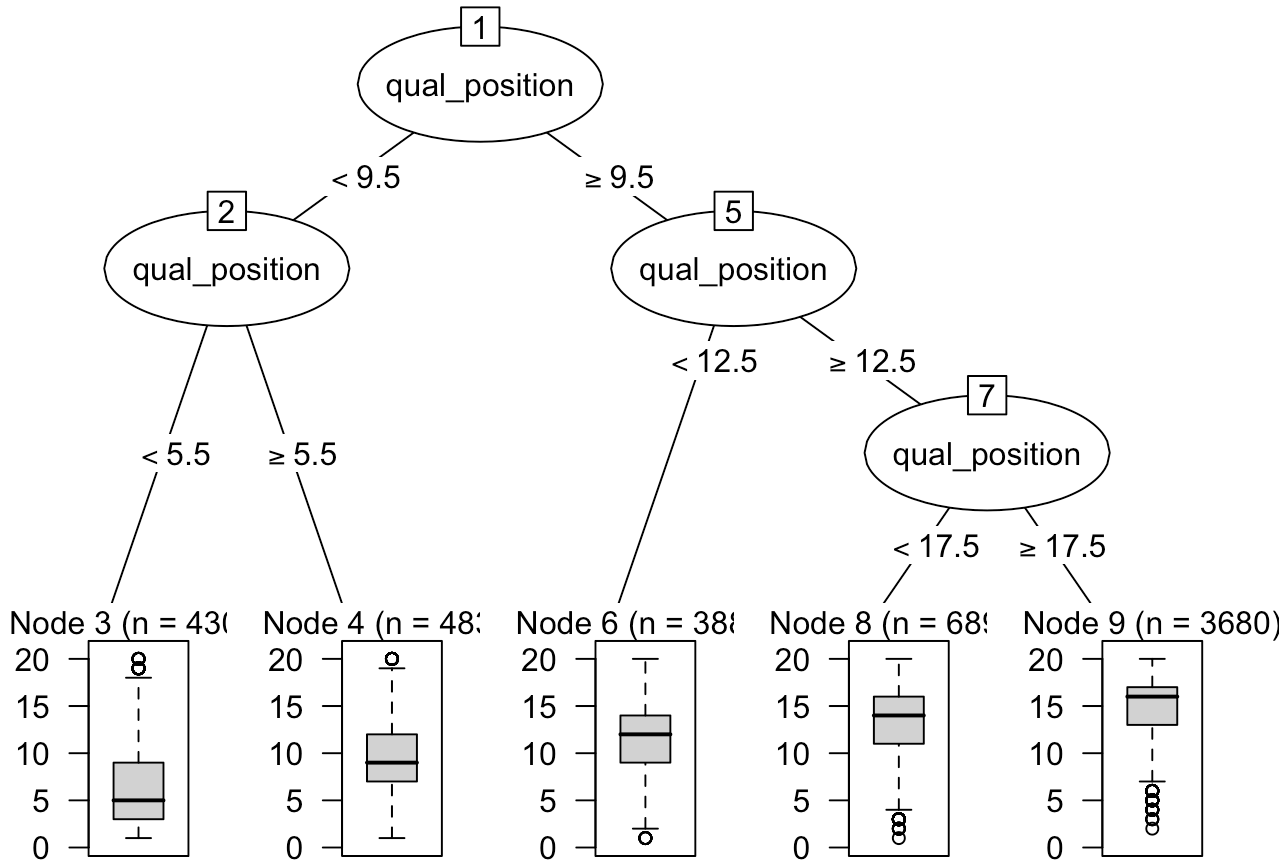
```
#Test
accuracy(test$pred_wqual,test$race_position)
```

```
##                    ME   RMSE      MAE       MPE     MAPE
## Test set -0.00467727 3.5437 2.816788 -20.30624 38.85112
```

```
accuracy(test$pred,test$race_position)
```

```
##                   ME     RMSE      MAE       MPE     MAPE
## Test set 0.01578868 4.062522 3.341963 -27.29216 48.74729
```

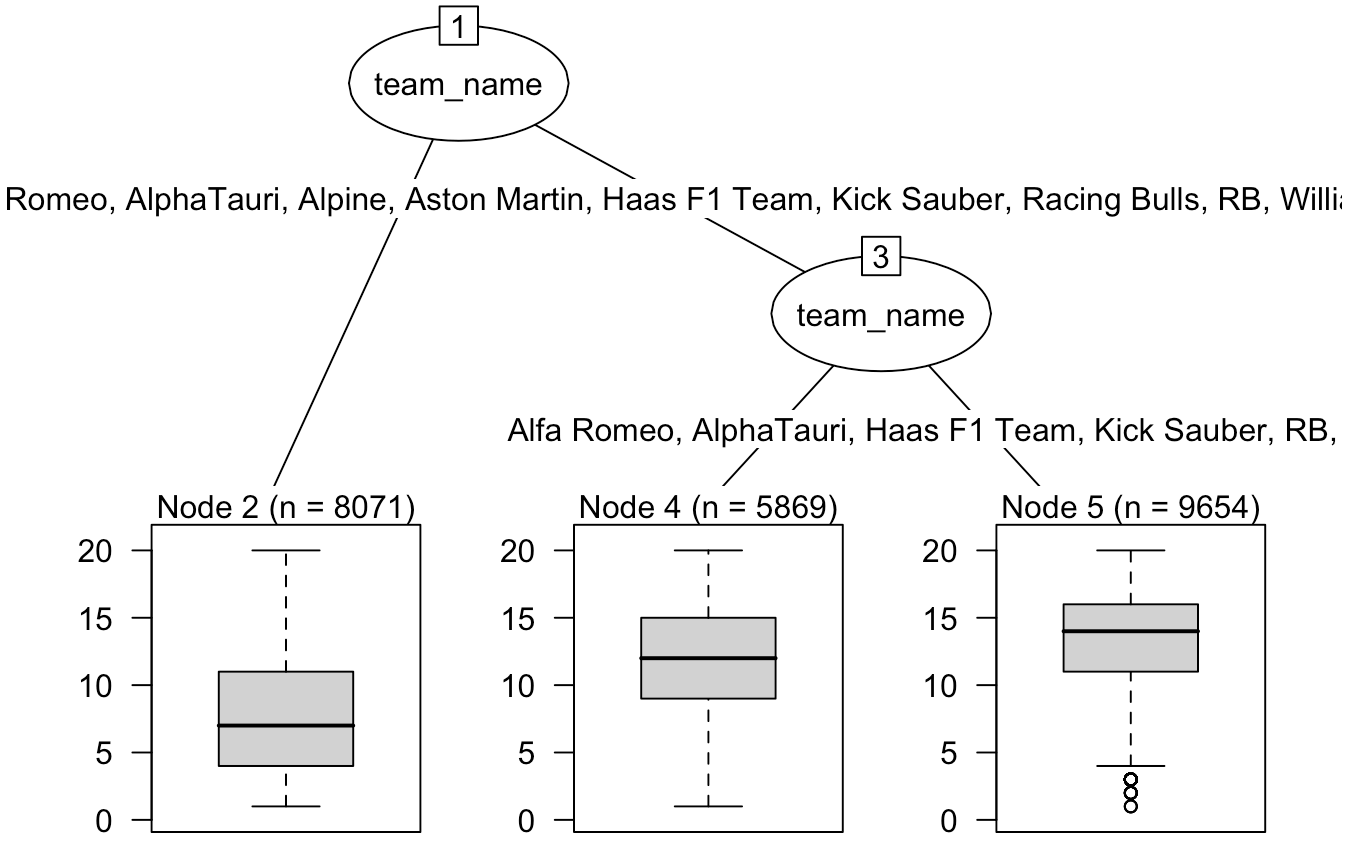
```
#Pretty Good MAE

# ------------------------
# Decision Tree
# ------------------------
#With qual
reg_tree_wqual <- rpart(race_position ~ ., data = train[c("team_name","wind_speed","rain
fall","humidity","pressure","qual_position","loc_cluster","temp_diff","race_position")])

plot(as.party(reg_tree_wqual), type="extended")
```




```
printcp(reg_tree_wqual)
```

```
##
## Regression tree:
## rpart(formula = race_position ~ ., data = train[c("team_name",
##     "wind_speed", "rainfall", "humidity", "pressure", "qual_position",
##     "loc_cluster", "temp_diff", "race_position")])
##
## Variables actually used in tree construction:
## [1] qual_position
##
## Root node error: 521840/23594 = 22.117
##
## n= 23594
##
##         CP nsplit rel error  xerror      xstd
## 1 0.321565      0   1.00000 1.00007 0.0069094
## 2 0.035901      1   0.67844 0.67858 0.0059674
## 3 0.031257      2   0.64253 0.64571 0.0060100
## 4 0.011380      3   0.61128 0.61644 0.0059369
## 5 0.010000      4   0.59990 0.60561 0.0058975
```

```
#Somewhat useless as we see that qualifying position is fairly useless

#No qual
reg_tree <- rpart(race_position ~ ., data = train[c("team_name","wind_speed","rainfal
l","humidity","pressure","loc_cluster","temp_diff","race_position")])
plot(as.party(reg_tree), type="extended")
```




```
printcp(reg_tree)
```

```
##
## Regression tree:
## rpart(formula = race_position ~ ., data = train[c("team_name",
##     "wind_speed", "rainfall", "humidity", "pressure", "loc_cluster",
##     "temp_diff", "race_position")])
##
## Variables actually used in tree construction:
## [1] team_name
##
## Root node error: 521840/23594 = 22.117
##
## n= 23594
##
##         CP nsplit rel error  xerror      xstd
## 1 0.233892      0   1.00000 1.00004 0.0069092
## 2 0.012264      1   0.76611 0.76618 0.0062343
## 3 0.010000      2   0.75384 0.75394 0.0061682
```

```r
#This time only split on team name


# ------------------------
# Forecast Data for 2026
# ------------------------
forecast <- function(){
cities <- c(
  "Melbourne", "Shanghai", "Suzuka", "Sakhir", "Jeddah", "Miami",
  "Montreal", "Monaco", "Barcelona", "Spielberg", "Silverstone",
  "Spa-Francorchamps", "Budapest", "Zandvoort", "Monza", "Madrid",
  "Baku", "Singapore", "Austin", "Mexico City", "São Paulo",
  "Las Vegas", "Lusail", "Yas Marina"
)
start_dates_str <- c(
  "2026-03-06", "2026-03-13", "2026-03-27", "2026-04-10",
  "2026-04-17", "2026-05-01", "2026-05-22", "2026-06-05",
  "2026-06-12", "2026-06-26", "2026-07-03", "2026-07-17",
  "2026-07-24", "2026-08-21", "2026-09-04", "2026-09-11",
  "2026-09-24", "2026-10-09", "2026-10-23",
  "2026-10-30", "2026-11-06", "2026-11-19",
  "2026-11-27", "2026-12-04"
)
latitude <- c(
  -37.81, 31.23, 34.84, 26.06, 21.53, 25.76,
  45.50, 43.74, 41.39, 47.21, 52.09,
  50.44, 47.50, 52.38, 45.58, 40.42,
  40.41, 1.35, 30.27, 19.43, -23.70,
  36.17, 25.42, 24.48
)
longitude <- c(
  144.96, 121.47, 136.54, 50.53, 39.16, -80.19,
  -73.57, 7.42, 2.17, 14.80, -1.03,
  5.97, 19.04, 4.53, 9.27, -3.70,
  49.87, 103.82, -97.74, -99.13, -46.71,
  -115.14, 51.50, 54.61
)

race_dates_converted <- as.Date(start_dates_str)

races_2026 <- data.frame(
  lag_1 = race_dates_converted %m-% years(1),
  lag_2 = race_dates_converted %m-% years(2),
  lag_3 = race_dates_converted %m-% years(3),
  lag_4 = race_dates_converted %m-% years(4),
  lag_5 = race_dates_converted %m-% years(5),
  Latitude = latitude,
  Longitude = longitude,
  stringsAsFactors = FALSE
)
```

```
# -------------------------
# Import Future Weather Data
# -------------------------
  lag_weather <- rbind(
                      select(races_2026,begin_date = lag_1,Latitude,Longitude) %>%
                      pmap_df(.f=
                                function(begin_date,Latitude,Longitude){
                                  url <- paste0("https://archive-api.open-meteo.com/v1/a
rchive?latitude=",Latitude,"&longitude=",Longitude,"&start_date=",begin_date,"&end_date
=",begin_date,"&hourly=temperature_2m,relative_humidity_2m,precipitation,soil_temperatur
e_0_to_7cm,wind_speed_10m,surface_pressure")
                                  df <- fromJSON(content(GET(url), "text", encoding = "U
TF-8"))

                                  hourly_df <- df$hourly
                                  hourly_df$longitude <- Longitude
                                  hourly_df$latitude <- Latitude
                                  return(hourly_df)
                                }),
                        select(races_2026,begin_date = lag_2,Latitude,Longitude) %>%
                          pmap_df(.f=
                                    function(begin_date,Latitude,Longitude){
                                      url <- paste0("https://archive-api.open-meteo.com/
v1/archive?latitude=",Latitude,"&longitude=",Longitude,"&start_date=",begin_date,"&end_d
ate=",begin_date,"&hourly=temperature_2m,relative_humidity_2m,precipitation,soil_tempera
ture_0_to_7cm,wind_speed_10m,surface_pressure")
                                      df <- fromJSON(content(GET(url), "text", encoding
= "UTF-8"))

                                      hourly_df <- df$hourly
                                      hourly_df$longitude <- Longitude
                                      hourly_df$latitude <- Latitude
                                      return(hourly_df)
                                    }),
                        select(races_2026,begin_date = lag_3,Latitude,Longitude) %>%
                          pmap_df(.f=
                                    function(begin_date,Latitude,Longitude){
                                      url <- paste0("https://archive-api.open-meteo.com/
v1/archive?latitude=",Latitude,"&longitude=",Longitude,"&start_date=",begin_date,"&end_d
ate=",begin_date,"&hourly=temperature_2m,relative_humidity_2m,precipitation,soil_tempera
ture_0_to_7cm,wind_speed_10m,surface_pressure")
                                      df <- fromJSON(content(GET(url), "text", encoding
= "UTF-8"))

                                      hourly_df <- df$hourly
                                      hourly_df$longitude <- Longitude
                                      hourly_df$latitude <- Latitude
                                      return(hourly_df)
                                    }),
                        select(races_2026,begin_date = lag_4,Latitude,Longitude) %>%
                          pmap_df(.f=
                                    function(begin_date,Latitude,Longitude){
```

```
                                    url <- paste0("https://archive-api.open-meteo.com/
v1/archive?latitude=",Latitude,"&longitude=",Longitude,"&start_date=",begin_date,"&end_d
ate=",begin_date,"&hourly=temperature_2m,relative_humidity_2m,precipitation,soil_tempera
ture_0_to_7cm,wind_speed_10m,surface_pressure")
                                    df <- fromJSON(content(GET(url), "text", encoding
= "UTF-8"))

                                    hourly_df <- df$hourly
                                    hourly_df$longitude <- Longitude
                                    hourly_df$latitude <- Latitude
                                    return(hourly_df)
                                  }),
                        select(races_2026,begin_date = lag_5,Latitude,Longitude) %>%
                          pmap_df(.f=
                                    function(begin_date,Latitude,Longitude){
                                    url <- paste0("https://archive-api.open-meteo.com/
v1/archive?latitude=",Latitude,"&longitude=",Longitude,"&start_date=",begin_date,"&end_d
ate=",begin_date,"&hourly=temperature_2m,relative_humidity_2m,precipitation,soil_tempera
ture_0_to_7cm,wind_speed_10m,surface_pressure")
                                    df <- fromJSON(content(GET(url), "text", encoding
= "UTF-8"))

                                    hourly_df <- df$hourly
                                    hourly_df$longitude <- Longitude
                                    hourly_df$latitude <- Latitude
                                    return(hourly_df)
                                  }))

temp <- lag_weather
lag_weather <- temp


lag_weather <-  lag_weather[-1] %>%
                group_by(longitude,latitude) %>%
                summarise(temp_diff = mean(soil_temperature_0_to_7cm) - mean(temperature
_2m),
                          humidity = mean(relative_humidity_2m),
                          rainfall = mean(precipitation),
                          wind_speed = mean(wind_speed_10m),
                          pressure = mean(surface_pressure)) %>%
                          filter(latitude == max(latitude))




# ------------------------
# Full 2026 Schedule
# ------------------------
schedule_2026 <- data.frame(location = cities, latitude, longitude) %>%
                 left_join(lag_weather,join_by(longitude,latitude))
schedule_2026 <- schedule_2026[c(-2,-3)]
```

```
teams <- unique(analysis_set$team_name)

forecast_2026 <- data.frame()
for (t in teams) {
  temp <- schedule_2026
  temp$team_name <- t
  forecast_2026 <- rbind(forecast_2026,temp)
}
return(forecast_2026)
}
forecast_2026 <- forecast()
```

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

```
#write.csv(forecast_2026,"2026_projected_data")
#forecast_2026 <- read.csv("2026_projected_data")
#Shiv

##
#Add Location Cluster
##
forecast_2026 <- left_join(forecast_2026,cluster_assignments,by="location") %>%
select("team_name","wind_speed","rainfall","humidity","pressure",loc_cluster="Track_Clus
ter","temp_diff","location")

# ##
#Note Montreal Madrid Singapore are new so they do not have a location cluster
#Will add these in
# ##
centroids <- final_clusters$centers
missing_clust <- forecast_2026[is.na(forecast_2026$loc_cluster),] %>%
                select("wind_speed",
                       "rainfall",
                       "humidity",
                       "pressure",
                       "temp_diff",
                       "location") %>%
                distinct(location, .keep_all = TRUE)

missing_clust$dist1 <-   sqrt((missing_clust$wind_speed - centroids[1,1])^2 +
                           (missing_clust$rainfall - centroids[1,2])^2 +
                           (missing_clust$humidity - centroids[1,5])^2 +
                           (missing_clust$pressure - centroids[1,6])^2 +
                           (missing_clust$temp_diff - centroids[1,7])^2)
missing_clust$dist2 <-   sqrt((missing_clust$wind_speed - centroids[2,1])^2 +
                              (missing_clust$rainfall - centroids[2,2])^2 +
                              (missing_clust$humidity - centroids[2,5])^2 +
                              (missing_clust$pressure - centroids[2,6])^2 +
                              (missing_clust$temp_diff - centroids[2,7])^2)
missing_clust$loc_cluster <- if_else(missing_clust$dist1 <= missing_clust$dist2,1,2)

missing_clust <- select(missing_clust,"loc_cluster","location")

forecast_2026 <- left_join(forecast_2026,missing_clust,by="location")
forecast_2026$loc_cluster <- if_else(!is.na(forecast_2026$loc_cluster.x),forecast_2026$l
oc_cluster.x,forecast_2026$loc_cluster.y)
forecast_2026 <- select(forecast_2026,-"loc_cluster.x",-"loc_cluster.y")


# ##
#Make forecast
# ##
forecast_2026$team_name <- as.factor(forecast_2026$team_name)
forecast_2026$loc_cluster <- as.factor(forecast_2026$loc_cluster)

forecast_2026$predicted_position <- predict(lm_race_step,forecast_2026)
```

```
standing_table <- group_by(forecast_2026,team_name) %>%
                  summarise(pos_sum = sum(predicted_position))

standing_table$predicted_rank <- rank(standing_table$pos_sum)
```