CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DE DADOS (BIG DATA PROCESSING AND ANALYTICS)

Componente curricular: PARADIGMAS DE LINGUAGEM DE PROGRAMAÇÃO EM CIÊNCIA DE DADOS [TURMA 01D] - 2022/1 - Trilha 7.

Aluno: ROBSON DE FREITAS SAMPAIO.

URL deste notebook: https://github.com/rfsampaio/postgraduate_data_science/blob/main/notebooks/PL_A7.ipynb

---

Exercício de Aprofundamento - Trilha 7

Faça as manipulações e explorações visuais de acordo com as perguntas que precisam ser respondidas

Análise de dados da NFL

Pacote do R: https://cran.r-project.org/web/packages/nflfastR/index.html

Este pacote permite que dados da NFL sejam analisados, jogada a jogada, habilitando diversos tipos de tomada de decisão a partir de manipulação dos dados e geração de gráficos.

Nesta atividade de aprofundamento, vamos explorar itens estudados tanto na trilha 6 com o pacote **Tidyverse** quanto na trilha 7 com o pacote **ggplot2**.

Algumas partes desta atividade já estão prontas, como por exemplo, o carregamento do conjunto de dados geral, a impressão dos escudos dos times e a segmentação de sub-conjuntos de dados para permitir uma manipulação mais simples na atividade.

Começamos então, com a instalação do pacote *nflfastR* e os carregamentos dos pacotes necessários.

In [ ]:
```
#install.packages("nflfastR")
#install.packages("ggimage")
#install.packages("imager")
```

In [ ]:
```
library(nflfastR)
library(tidyverse)
library(ggplot2)
library(imager)
#library(ggimage)
```

```
── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

✓ ggplot2 3.3.6     ✓ purrr   0.3.4
✓ tibble  3.1.7     ✓ dplyr   1.0.9
✓ tidyr   1.2.0     ✓ stringr 1.4.0
✓ readr   2.1.2     ✓ forcats 0.5.1

── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✕ dplyr::filter() masks stats::filter()
✕ dplyr::lag()    masks stats::lag()

Carregando pacotes exigidos: magrittr


Attaching package: 'magrittr'


The following object is masked from 'package:purrr':

    set_names


The following object is masked from 'package:tidyr':

    extract



Attaching package: 'imager'


The following object is masked from 'package:magrittr':

    add


The following object is masked from 'package:stringr':

    boundary


The following object is masked from 'package:tidyr':
```

```
    fill


The following objects are masked from 'package:stats':

    convolve, spectrum


The following object is masked from 'package:graphics':

    frame


The following object is masked from 'package:base':

    save.image
```

Como este pacote permite baixar dados de todas as temporadas, jogada a jogada, desde 1999, faremos um recorte apenas de 2014. A escolha deste ano foi aleatória, mesmo que possa parecer que foi escolhido de forma proposital por ser ultimo ano no qual *Seattle Seahawks* ganhou o *Super Bowl* (que é o jogo final da temporada e define o vencedor do campeonato). Fique a vontade para escolher qualquer outro ano, caso deseje estudar.

> Contudo, para este exercício de aprofundamento, **mantenha o ano de 2014**.

In [ ]:
```
temporada <- load_pbp(2014) #Carregamento dos dados, jogada a jogada, de 2014
```

Repare que para a seleção do subconjunto de dados, foi informado o ano da temporada desejado.

Poderiam ser um intervalo de outros anos, para isso, seria necessário definir o valor como **anoInicio:anoFim**, por exemplo: 2014:2018 e neste caso os dados seriam de 2014 até 2018.

> temporada <- load_pbp(2014:2018)

Repare que este conjunto de dados de pbp (*play-by-play* -- jogada a jogada) possui muitas variáveis. Ao chamar a função *names* colocando o nome do conjunto de dados, são retornadas todas as variáveis. Execute o bloco abaixo e conheça quais são estas variáveis.

In [ ]:
```
names(temporada)
```

'play_id' · 'game_id' · 'old_game_id' · 'home_team' · 'away_team' · 'season_type' · 'week' · 'posteam' · 'posteam_type' · 'defteam' · 'side_of_field' · 'yardline_100' · 'game_date' · 'quarter_seconds_remaining' · 'half_seconds_remaining' · 'game_seconds_remaining' · 'game_half' · 'quarter_end' · 'drive' · 'sp' · 'qtr' · 'down' · 'goal_to_go' · 'time' · 'yrdln' · 'ydstogo' · 'ydsnet' · 'desc' · 'play_type' · 'yards_gained' · 'shotgun' · 'no_huddle' · 'qb_dropback' · 'qb_kneel' · 'qb_spike' · 'qb_scramble' · 'pass_length' · 'pass_location' · 'air_yards' · 'yards_after_catch' · 'run_location' · 'run_gap' · 'field_goal_result' · 'kick_distance' · 'extra_point_result' · 'two_point_conv_result' · 'home_timeouts_remaining' · 'away_timeouts_remaining' · 'timeout' · 'timeout_team' · 'td_team' · 'td_player_name' · 'td_player_id' · 'posteam_timeouts_remaining' · 'defteam_timeouts_remaining' · 'total_home_score' · 'total_away_score' · 'posteam_score' · 'defteam_score' · 'score_differential' · 'posteam_score_post' · 'defteam_score_post' · 'score_differential_post' · 'no_score_prob' · 'opp_fg_prob' · 'opp_safety_prob' · 'opp_td_prob' · 'fg_prob' · 'safety_prob' · 'td_prob' · 'extra_point_prob' · 'two_point_conversion_prob' · 'ep' · 'epa' · 'total_home_epa' · 'total_away_epa' · 'total_home_rush_epa' · 'total_away_rush_epa' · 'total_home_pass_epa' · 'total_away_pass_epa' · 'air_epa' · 'yac_epa' · 'comp_air_epa' · 'comp_yac_epa' · 'total_home_comp_air_epa' · 'total_away_comp_air_epa' · 'total_home_comp_yac_epa' · 'total_away_comp_yac_epa' · 'total_home_raw_air_epa' · 'total_away_raw_air_epa' · 'total_home_raw_yac_epa' · 'total_away_raw_yac_epa' · 'wp' · 'def_wp' · 'home_wp' · 'away_wp' · 'wpa' · 'vegas_wpa' · 'vegas_home_wpa' · 'home_wp_post' · 'away_wp_post' · 'vegas_wp' · 'vegas_home_wp' · 'total_home_rush_wpa' · 'total_away_rush_wpa' · 'total_home_pass_wpa' · 'total_away_pass_wpa' · 'air_wpa' · 'yac_wpa' · 'comp_air_wpa' · 'comp_yac_wpa' · 'total_home_comp_air_wpa' · 'total_away_comp_air_wpa' · 'total_home_comp_yac_wpa' · 'total_away_comp_yac_wpa' · 'total_home_raw_air_wpa' · 'total_away_raw_air_wpa' · 'total_home_raw_yac_wpa' · 'total_away_raw_yac_wpa' · 'punt_blocked' · 'first_down_rush' · 'first_down_pass' · 'first_down_penalty' · 'third_down_converted' · 'third_down_failed' · 'fourth_down_converted' · 'fourth_down_failed' · 'incomplete_pass' · 'touchback' · 'interception' · 'punt_inside_twenty' · 'punt_in_endzone' · 'punt_out_of_bounds' · 'punt_downed' · 'punt_fair_catch' · 'kickoff_inside_twenty' · 'kickoff_in_endzone' · 'kickoff_out_of_bounds' · 'kickoff_downed' · 'kickoff_fair_catch' · 'fumble_forced' · 'fumble_not_forced' · 'fumble_out_of_bounds' · 'solo_tackle' · 'safety' · 'penalty' · 'tackled_for_loss' · 'fumble_lost' · 'own_kickoff_recovery' · 'own_kickoff_recovery_td' · 'qb_hit' · 'rush_attempt' · 'pass_attempt' · 'sack' · 'touchdown' · 'pass_touchdown' · 'rush_touchdown' · 'return_touchdown' · 'extra_point_attempt' · 'two_point_attempt' · 'field_goal_attempt' · 'kickoff_attempt' · 'punt_attempt' · 'fumble' · 'complete_pass' · 'assist_tackle' · 'lateral_reception' · 'lateral_rush' · 'lateral_return' · 'lateral_recovery' · 'passer_player_id' · 'passer_player_name' · 'passing_yards' · 'receiver_player_id' · 'receiver_player_name' · 'receiving_yards' · 'rusher_player_id' · 'rusher_player_name' · 'rushing_yards' · 'lateral_receiver_player_id' · 'lateral_receiver_player_name' · 'lateral_receiving_yards' · 'lateral_rusher_player_id' · 'lateral_rusher_player_name' · 'lateral_rushing_yards' · 'lateral_sack_player_id' · 'lateral_sack_player_name' · 'interception_player_id' · 'interception_player_name' · 'lateral_interception_player_id' · 'lateral_interception_player_name' · 'punt_returner_player_id' · 'punt_returner_player_name' · 'lateral_punt_returner_player_id' · 'lateral_punt_returner_player_name' · 'kickoff_returner_player_name' · 'kickoff_returner_player_id' · 'lateral_kickoff_returner_player_id' · 'lateral_kickoff_returner_player_name' · 'punter_player_id' · 'punter_player_name' · 'kicker_player_name' · 'kicker_player_id' · 'own_kickoff_recovery_player_id' · 'own_kickoff_recovery_player_name' · 'blocked_player_id' · 'blocked_player_name' · 'tackle_for_loss_1_player_id' · 'tackle_for_loss_1_player_name' · 'tackle_for_loss_2_player_id' · 'tackle_for_loss_2_player_name' · 'qb_hit_1_player_id' · 'qb_hit_1_player_name' · 'qb_hit_2_player_id' · 'qb_hit_2_player_name' ·

'forced_fumble_player_1_team' · 'forced_fumble_player_1_player_id' · 'forced_fumble_player_1_player_name' · 'forced_fumble_player_2_team' · 'forced_fumble_player_2_player_id' · 'forced_fumble_player_2_player_name' · 'solo_tackle_1_team' · 'solo_tackle_2_team' · 'solo_tackle_1_player_id' · 'solo_tackle_2_player_id' · 'solo_tackle_1_player_name' · 'solo_tackle_2_player_name' · 'assist_tackle_1_player_id' · 'assist_tackle_1_player_name' · 'assist_tackle_1_team' · 'assist_tackle_2_player_id' · 'assist_tackle_2_player_name' · 'assist_tackle_2_team' · 'assist_tackle_3_player_id' · 'assist_tackle_3_player_name' · 'assist_tackle_3_team' · 'assist_tackle_4_player_id' · 'assist_tackle_4_player_name' · 'assist_tackle_4_team' · 'tackle_with_assist' · 'tackle_with_assist_1_player_id' · 'tackle_with_assist_1_player_name' · 'tackle_with_assist_1_team' · 'tackle_with_assist_2_player_id' · 'tackle_with_assist_2_player_name' · 'tackle_with_assist_2_team' · 'pass_defense_1_player_id' · 'pass_defense_1_player_name' · 'pass_defense_2_player_id' · 'pass_defense_2_player_name' · 'fumbled_1_team' · 'fumbled_1_player_id' · 'fumbled_1_player_name' · 'fumbled_2_player_id' · 'fumbled_2_player_name' · 'fumbled_2_team' · 'fumble_recovery_1_team' · 'fumble_recovery_1_yards' · 'fumble_recovery_1_player_id' · 'fumble_recovery_1_player_name' · 'fumble_recovery_2_team' · 'fumble_recovery_2_yards' · 'fumble_recovery_2_player_id' · 'fumble_recovery_2_player_name' · 'sack_player_id' · 'sack_player_name' · 'half_sack_1_player_id' · 'half_sack_1_player_name' · 'half_sack_2_player_id' · 'half_sack_2_player_name' · 'return_team' · 'return_yards' · 'penalty_team' · 'penalty_player_id' · 'penalty_player_name' · 'penalty_yards' · 'replay_or_challenge' · 'replay_or_challenge_result' · 'penalty_type' · 'defensive_two_point_attempt' · 'defensive_two_point_conv' · 'defensive_extra_point_attempt' · 'defensive_extra_point_conv' · 'safety_player_name' · 'safety_player_id' · 'season' · 'cp' · 'cpoe' · 'series' · 'series_success' · 'series_result' · 'order_sequence' · 'start_time' · 'time_of_day' · 'stadium' · 'weather' · 'nfl_api_id' · 'play_clock' · 'play_deleted' · 'play_type_nfl' · 'special_teams_play' · 'st_play_type' · 'end_clock_time' · 'end_yard_line' · 'fixed_drive' · 'fixed_drive_result' · 'drive_real_start_time' · 'drive_play_count' · 'drive_time_of_possession' · 'drive_first_downs' · 'drive_inside20' · 'drive_ended_with_score' · 'drive_quarter_start' · 'drive_quarter_end' · 'drive_yards_penalized' · 'drive_start_transition' · 'drive_end_transition' · 'drive_game_clock_start' · 'drive_game_clock_end' · 'drive_start_yard_line' · 'drive_end_yard_line' · 'drive_play_id_started' · 'drive_play_id_ended' · 'away_score' · 'home_score' · 'location' · 'result' · 'total' · 'spread_line' · 'total_line' · 'div_game' · 'roof' · 'surface' · 'temp' · 'wind' · 'home_coach' · 'away_coach' · 'stadium_id' · 'game_stadium' · 'aborted_play' · 'success' · 'passer' · 'passer_jersey_number' · 'rusher' · 'rusher_jersey_number' · 'receiver' · 'receiver_jersey_number' · 'pass' · 'rush' · 'first_down' · 'special' · 'play' · 'passer_id' · 'rusher_id' · 'receiver_id' · 'name' · 'jersey_number' · 'id' · 'fantasy_player_name' · 'fantasy_player_id' · 'fantasy' · 'fantasy_id' · 'out_of_bounds' · 'home_opening_kickoff' · 'qb_epa' · 'xyac_epa' · 'xyac_mean_yardage' · 'xyac_median_yardage' · 'xyac_success' · 'xyac_fd' · 'xpass' · 'pass_oe'

Para conhecer os times que jogam na NFL, é possível ter um retorno de dados básicos dos cada um deles. Este retorno básico pode ser transformado em um data frame, para posteriormente, ser utilizado como filtro da estrutura.

In [ ]:
```
times <- teams_colors_logos %>% unique()
```

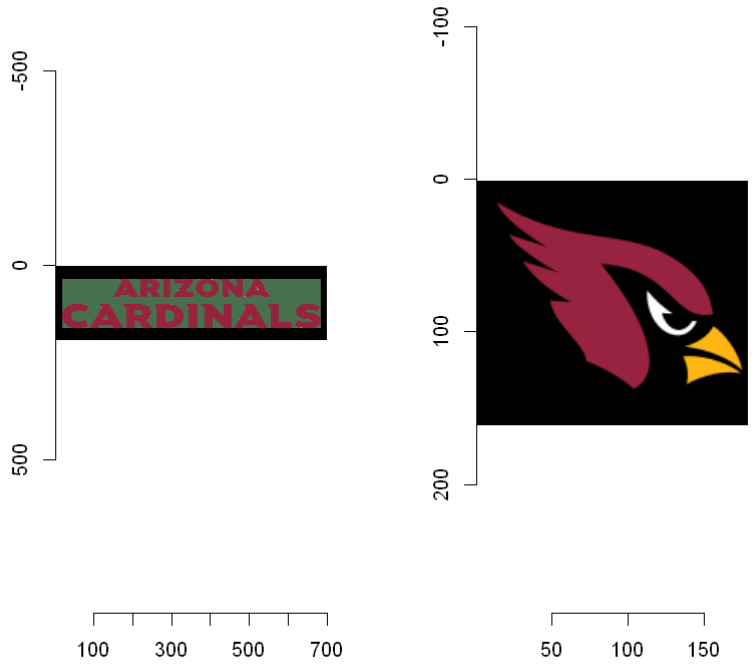
In [ ]:
```
names(times)
```

'team_abbr' · 'team_name' · 'team_id' · 'team_nick' · 'team_color' · 'team_color2' · 'team_color3' · 'team_color4' · 'team_logo_wikipedia' · 'team_logo_espn' · 'team_wordmark'

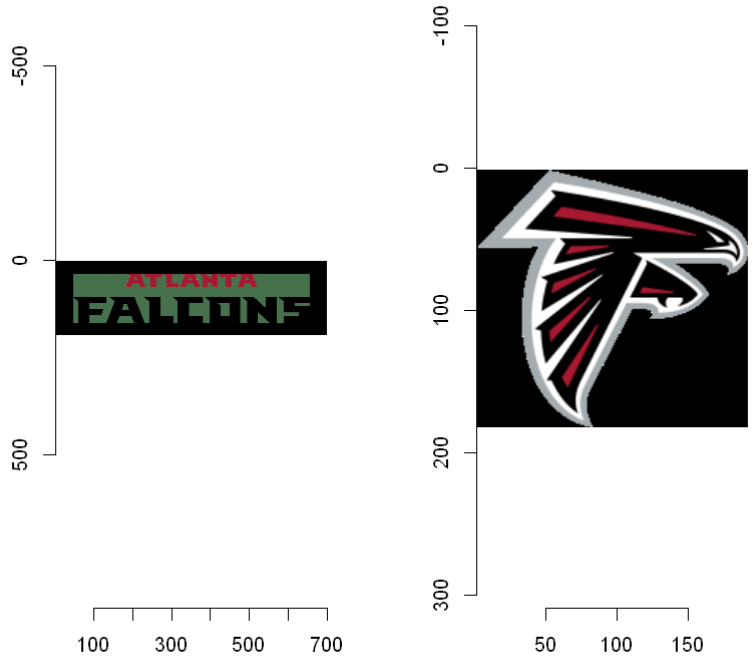
Aproveitando que estamos observando os times, é possível mostrar todos os seus escudos e nomes. Para isso, execute o bloco de codigo abaixo, e veja como é a saída:

In [ ]:
```
for (i in 1:dim(times)[1]){
    par(mfrow=c(1,2))
    load.image(as.character(times[i,'team_wordmark'])) %>% plot ;
    load.image(as.character(times[i,'team_logo_wikipedia'])) %>% plot ;
    print(paste(times[i,'team_name'],times[i,'team_abbr'],sep=' >> '));
    par(mfrow=c(1,1))
}
```

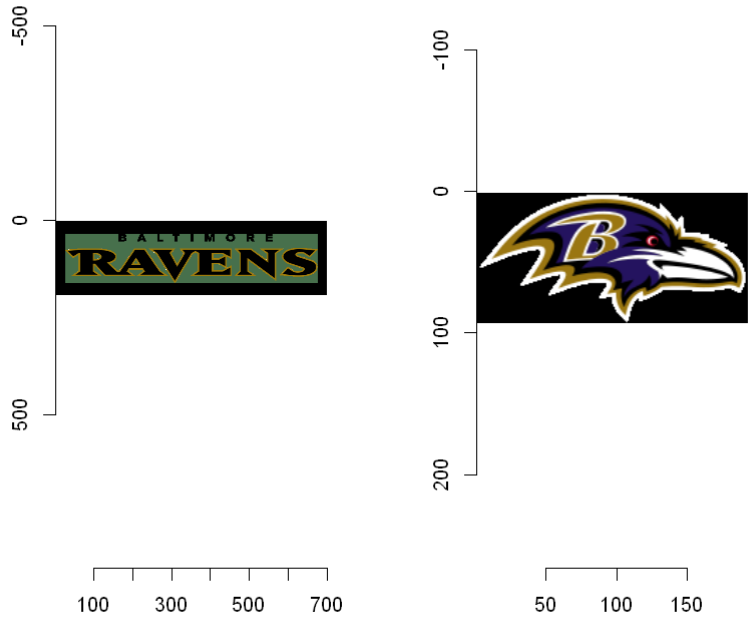
```
[1] "Arizona Cardinals >> ARI"
```



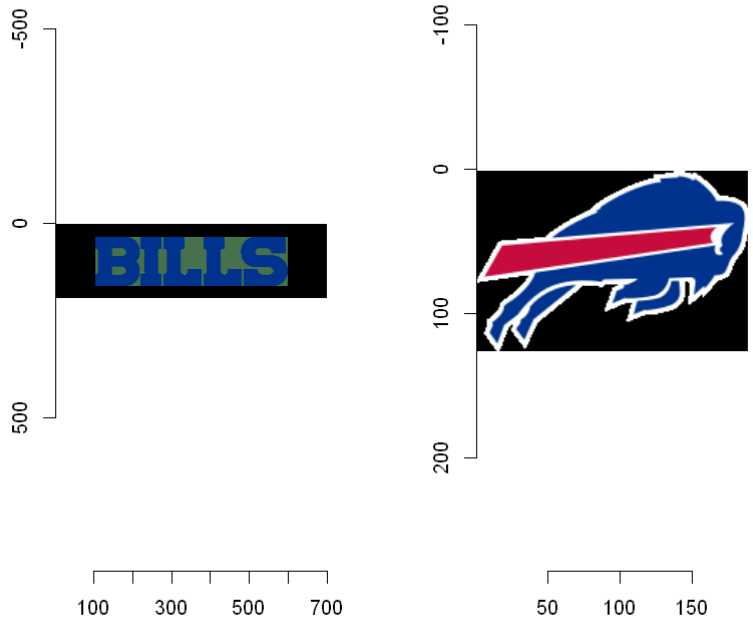

```
[1] "Atlanta Falcons >> ATL"
```



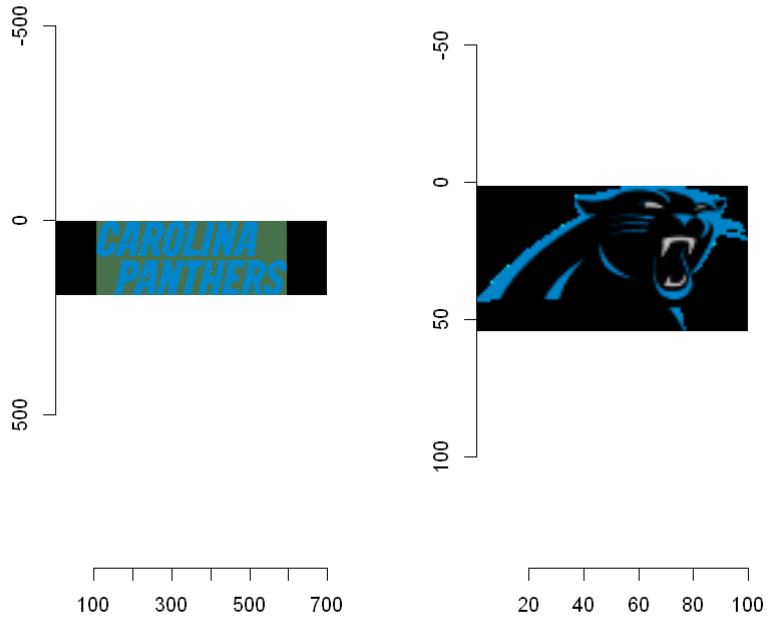

```
[1] "Baltimore Ravens >> BAL"
```



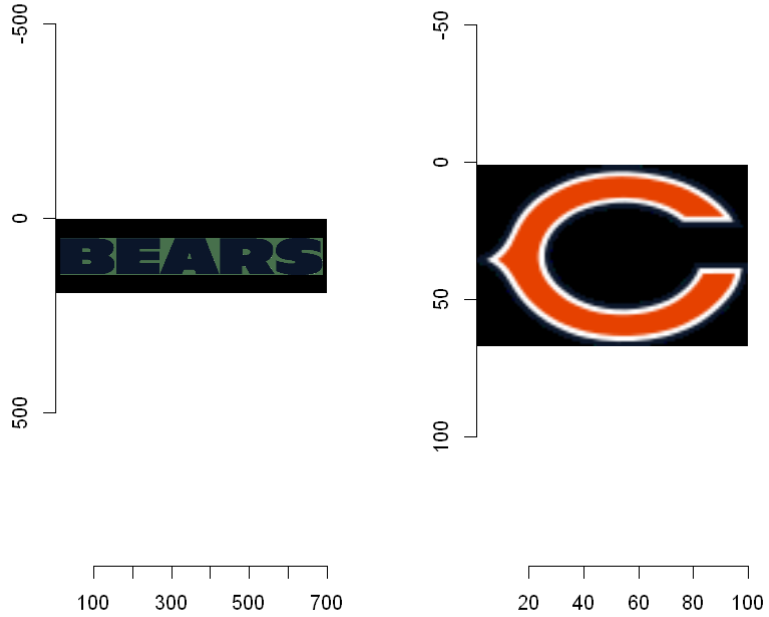

```
[1] "Buffalo Bills >> BUF"
```



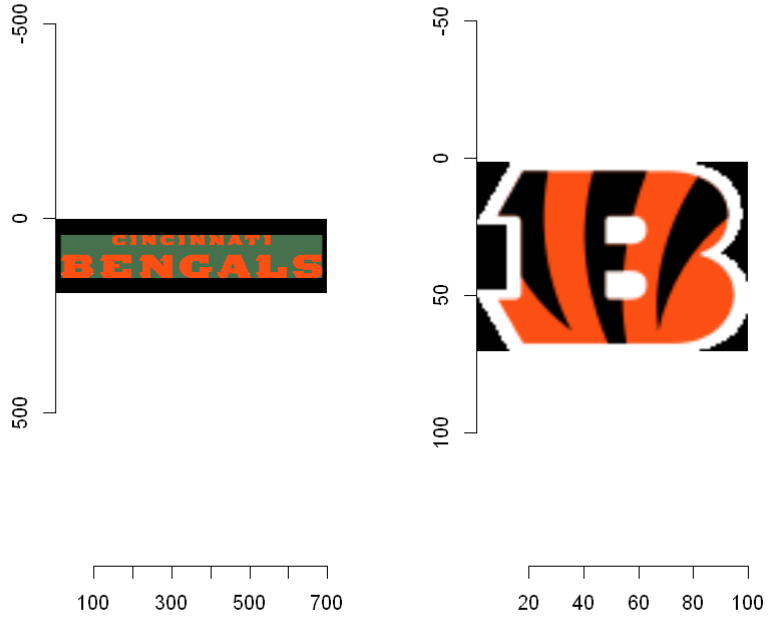

```
[1] "Carolina Panthers >> CAR"
```



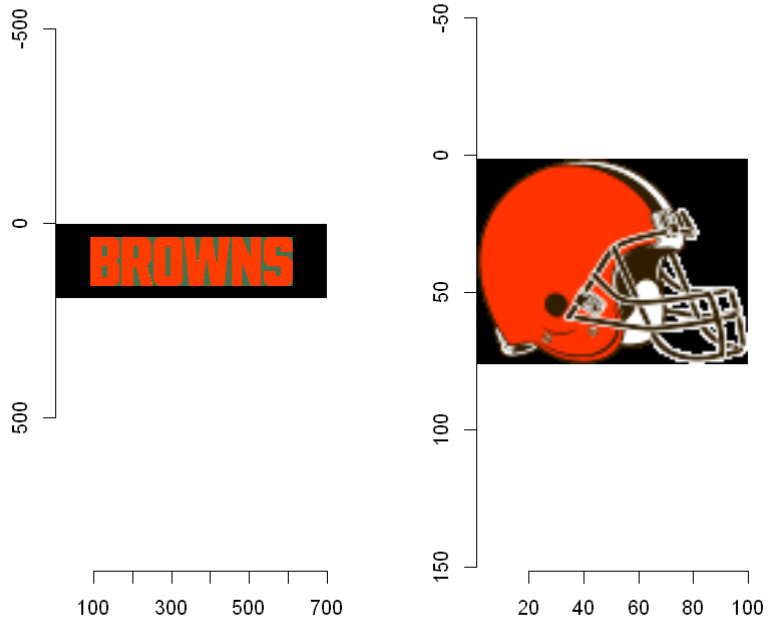

```
[1] "Chicago Bears >> CHI"
```



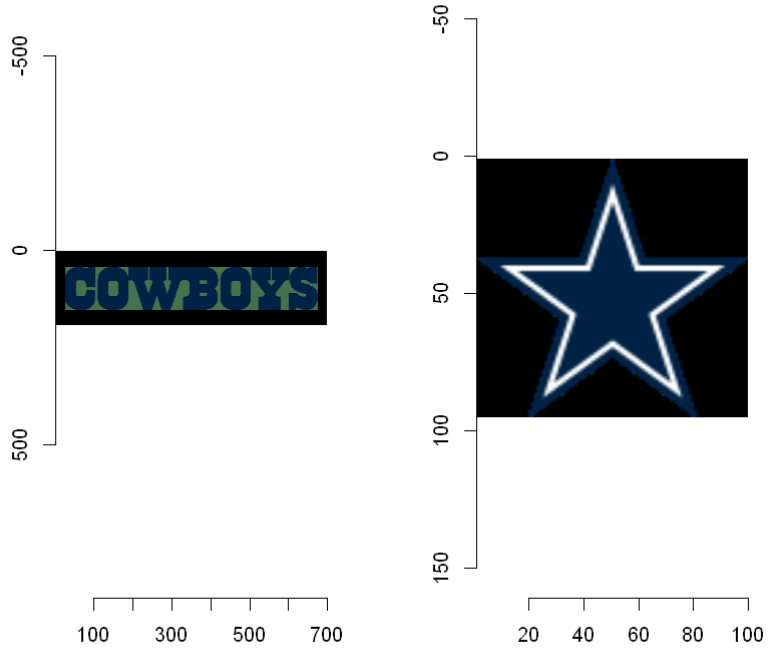

```
[1] "Cincinnati Bengals >> CIN"
```



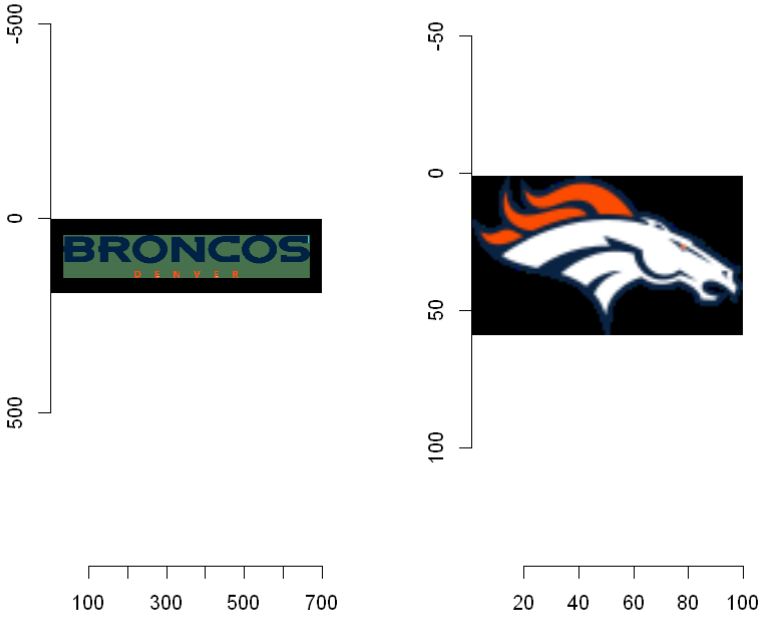

```
[1] "Cleveland Browns >> CLE"
```



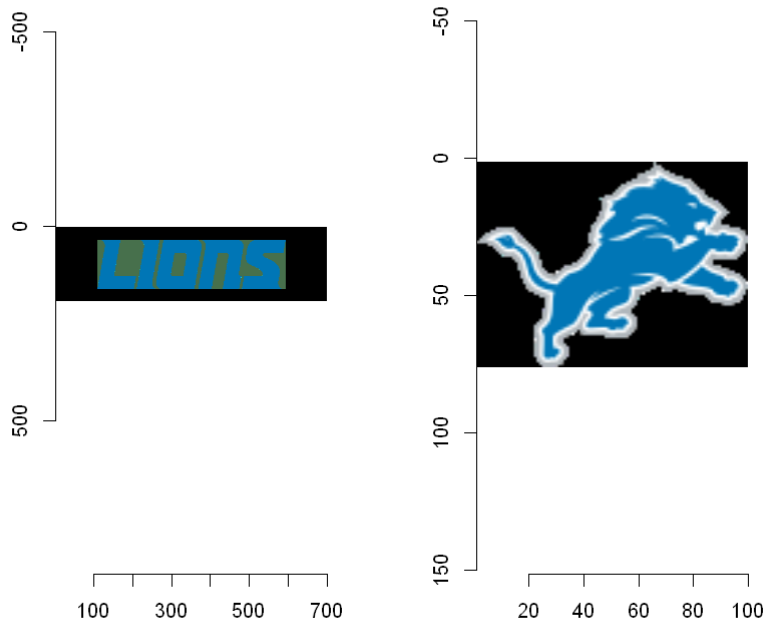

```
[1] "Dallas Cowboys >> DAL"
```



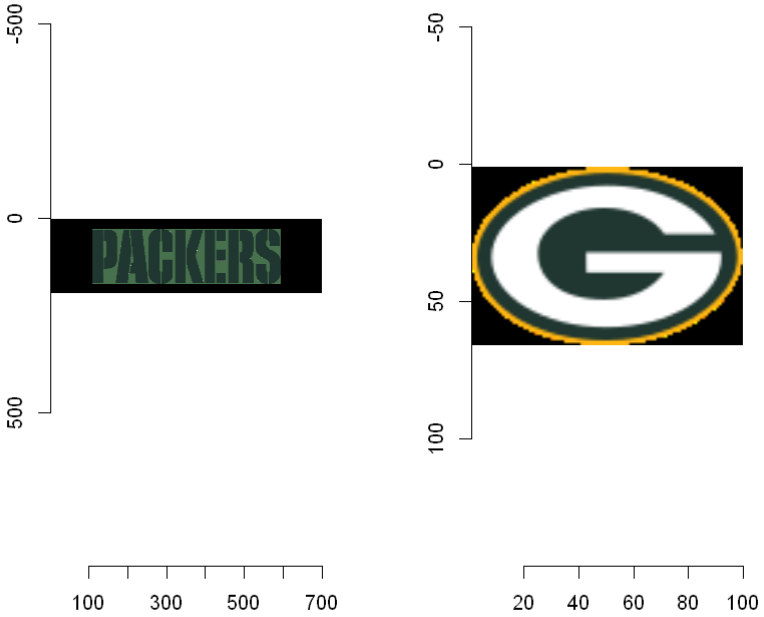

```
[1] "Denver Broncos >> DEN"
```



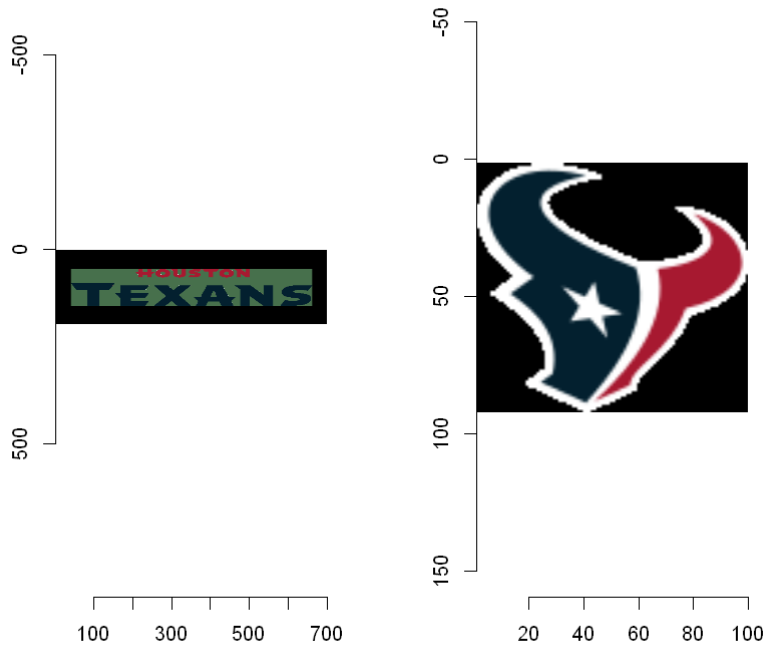

```
[1] "Detroit Lions >> DET"
```



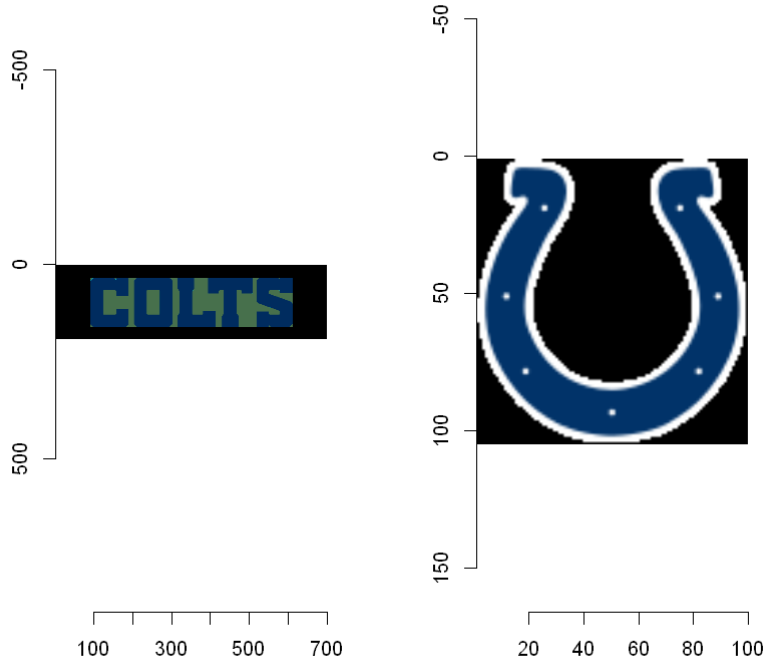

```
[1] "Green Bay Packers >> GB"
```



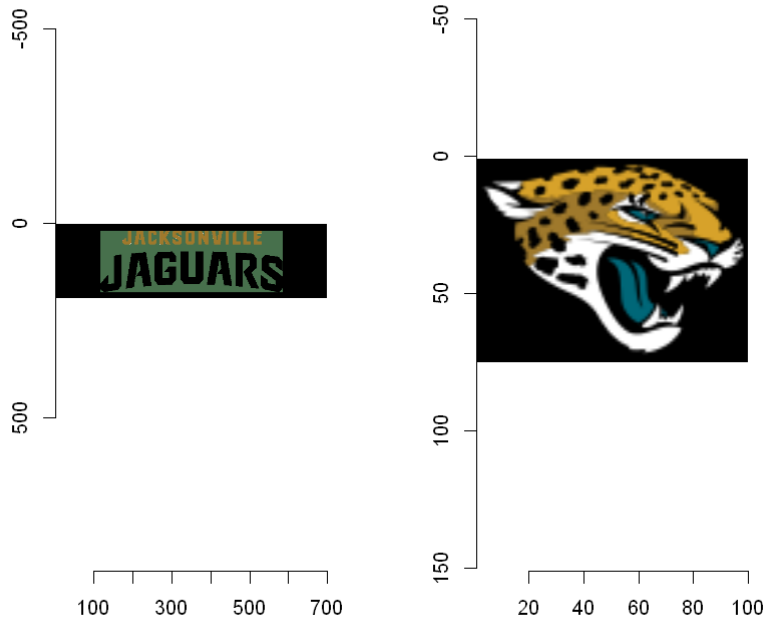

```
[1] "Houston Texans >> HOU"
```



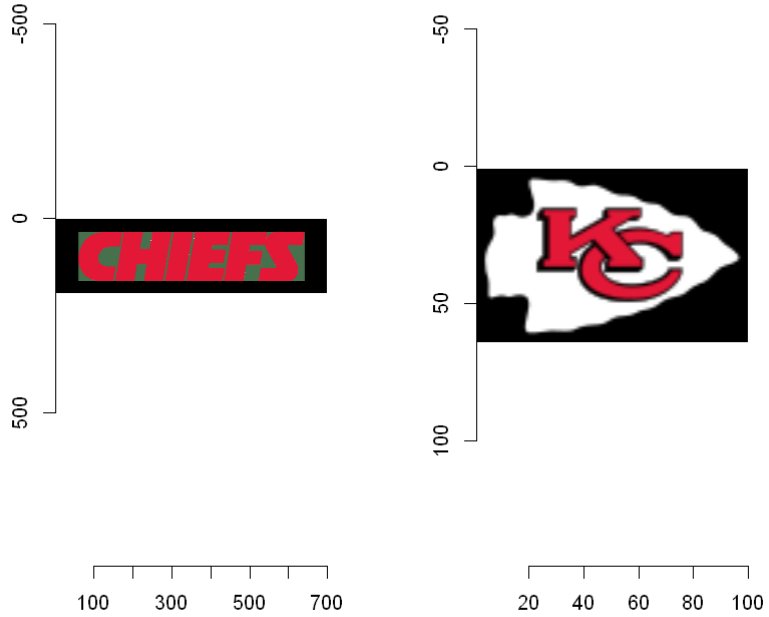

```
[1] "Indianapolis Colts >> IND"
```



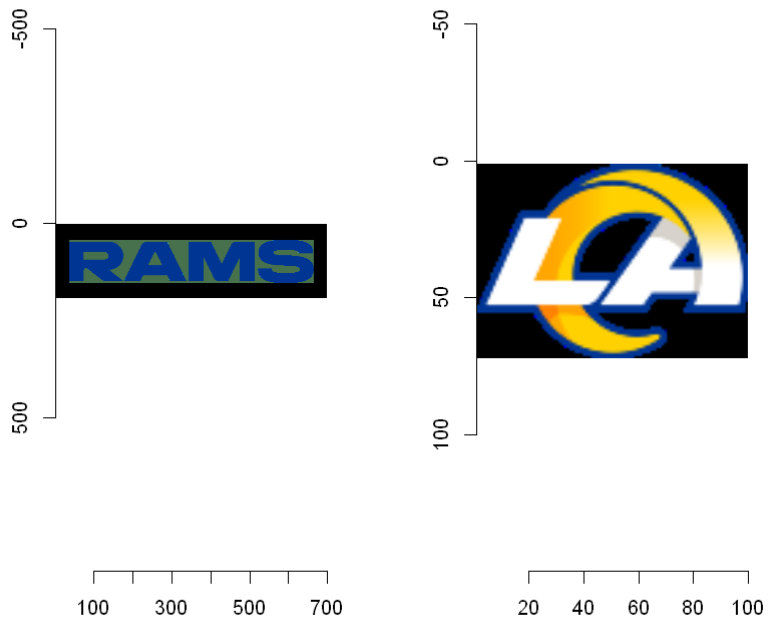

```
[1] "Jacksonville Jaguars >> JAX"
```



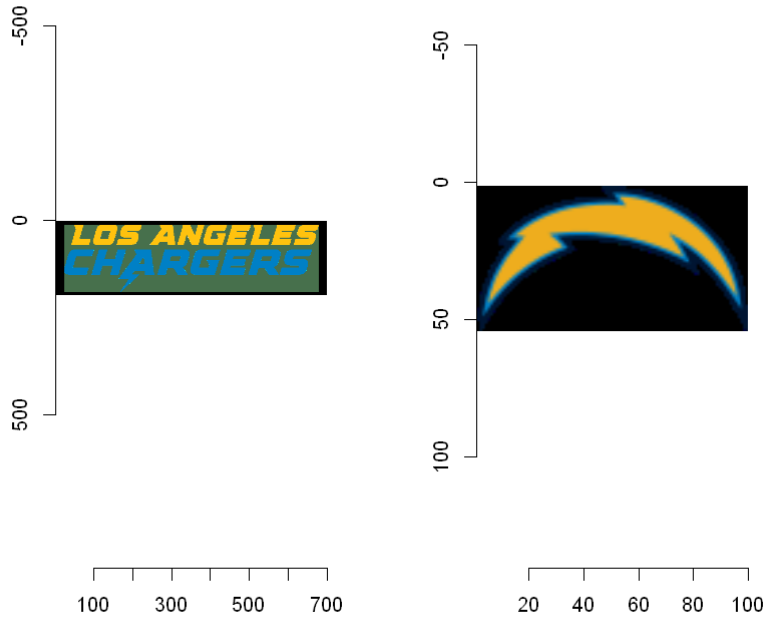

```
[1] "Kansas City Chiefs >> KC"
```



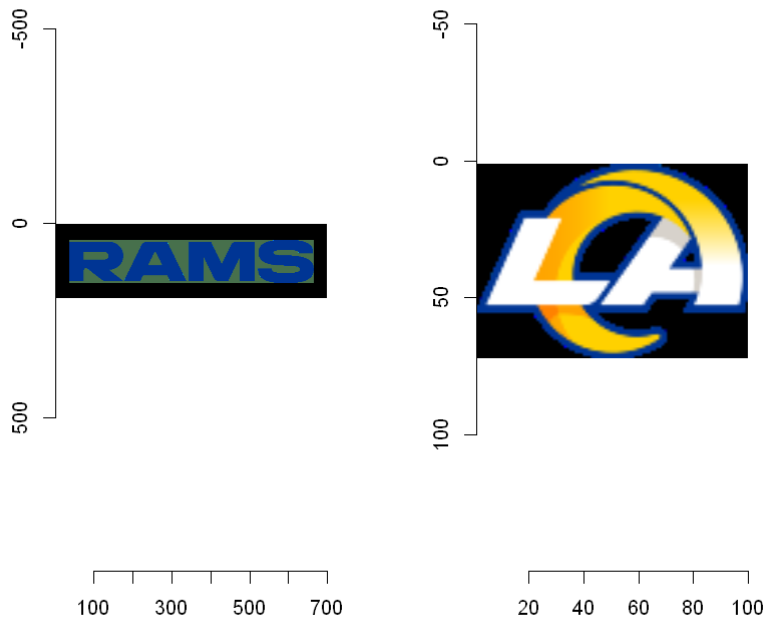

```
[1] "Los Angeles Rams >> LA"
```



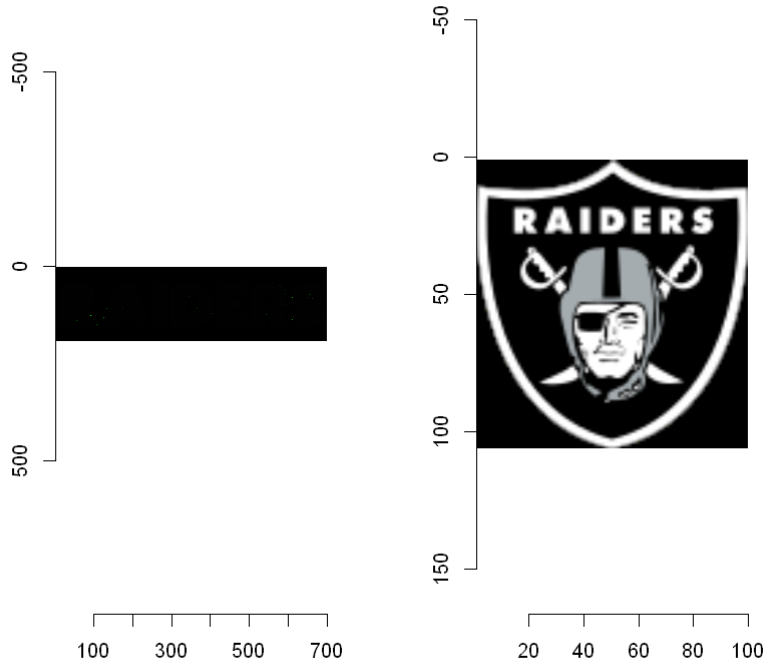

```
[1] "Los Angeles Chargers >> LAC"
```



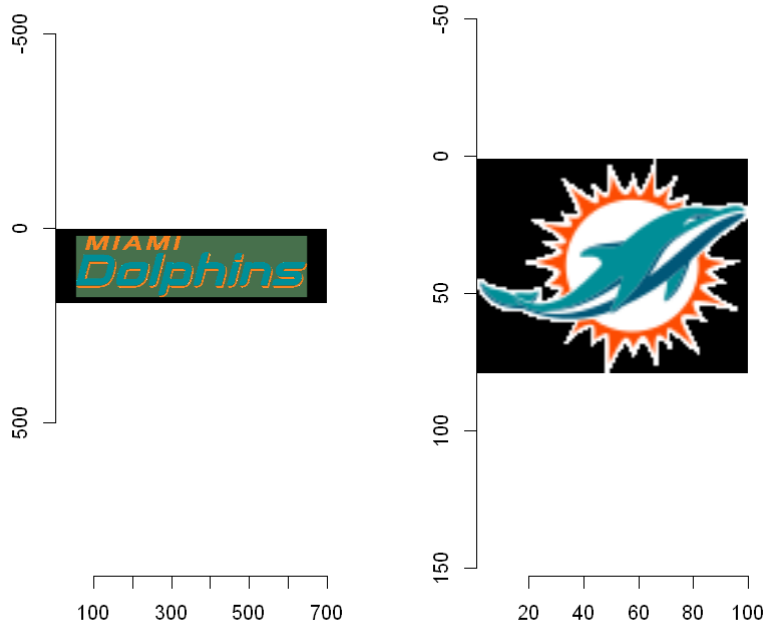

```
[1] "Los Angeles Rams >> LAR"
```



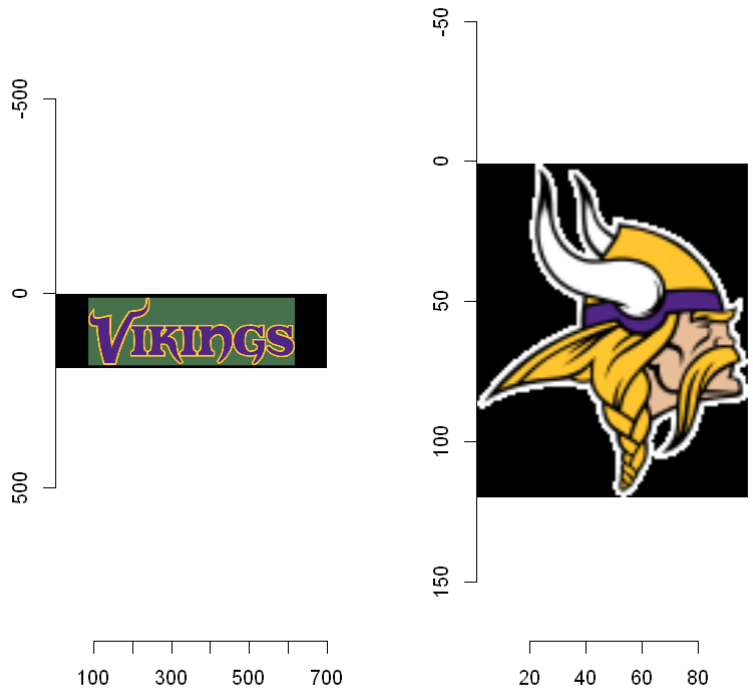

```
[1] "Las Vegas Raiders >> LV"
```



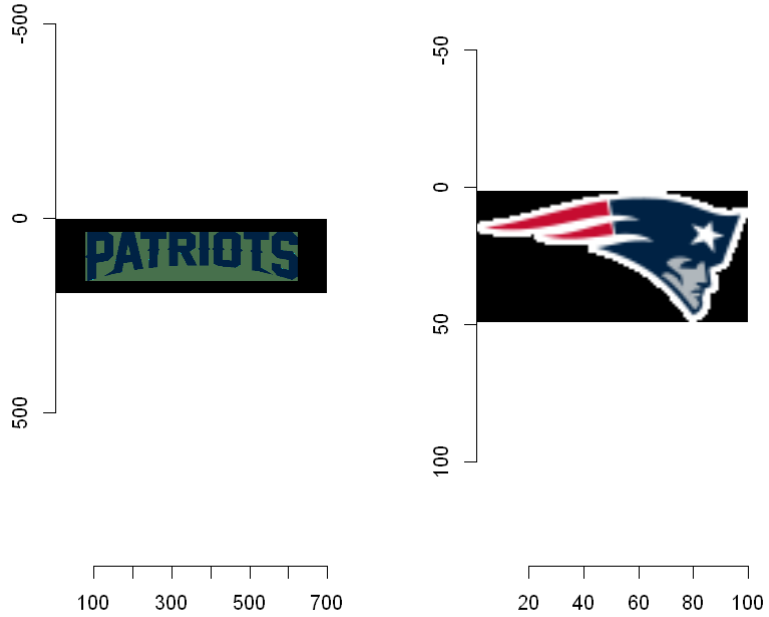

```
[1] "Miami Dolphins >> MIA"
```



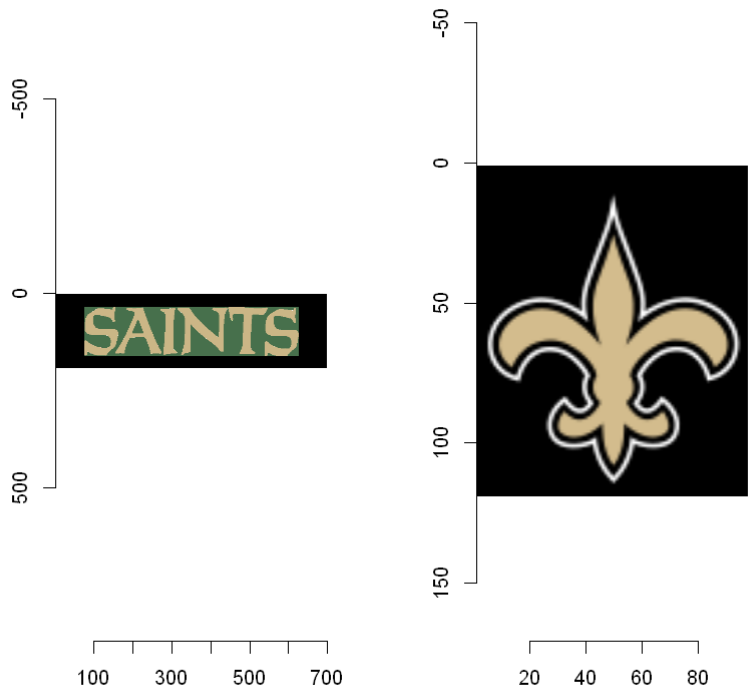

```
[1] "Minnesota Vikings >> MIN"
```



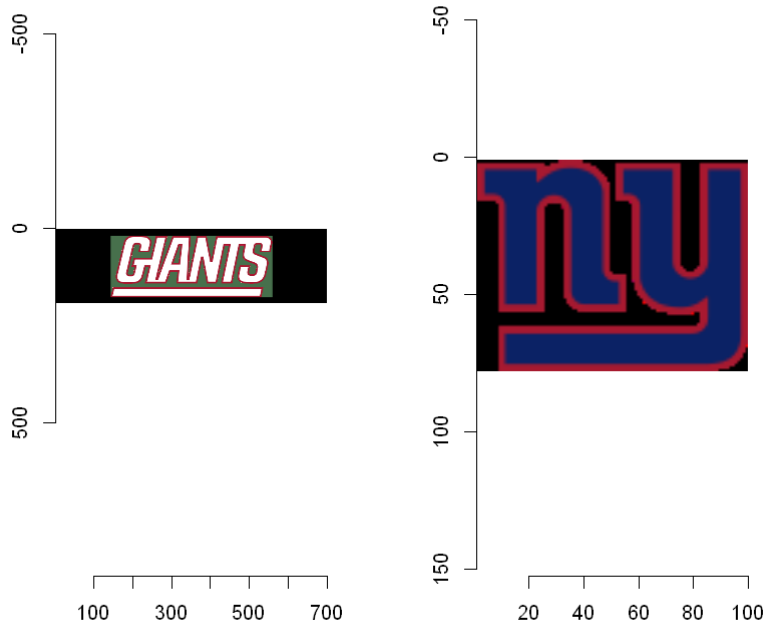

```
[1] "New England Patriots >> NE"
```



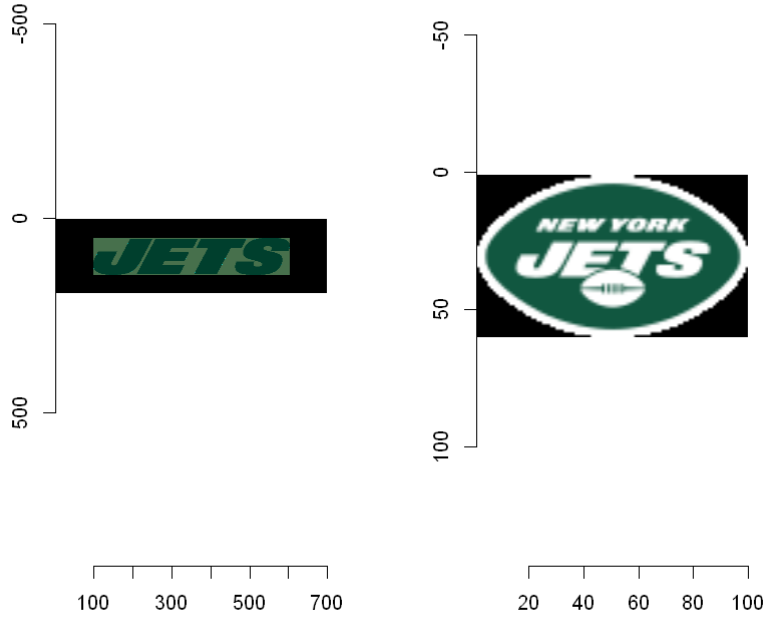

```
[1] "New Orleans Saints >> NO"
```



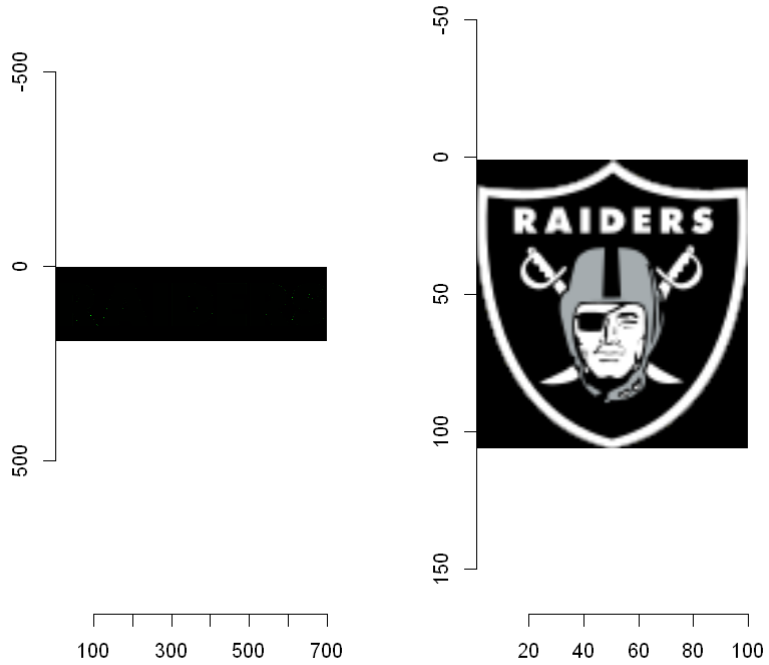

```
[1] "New York Giants >> NYG"
```



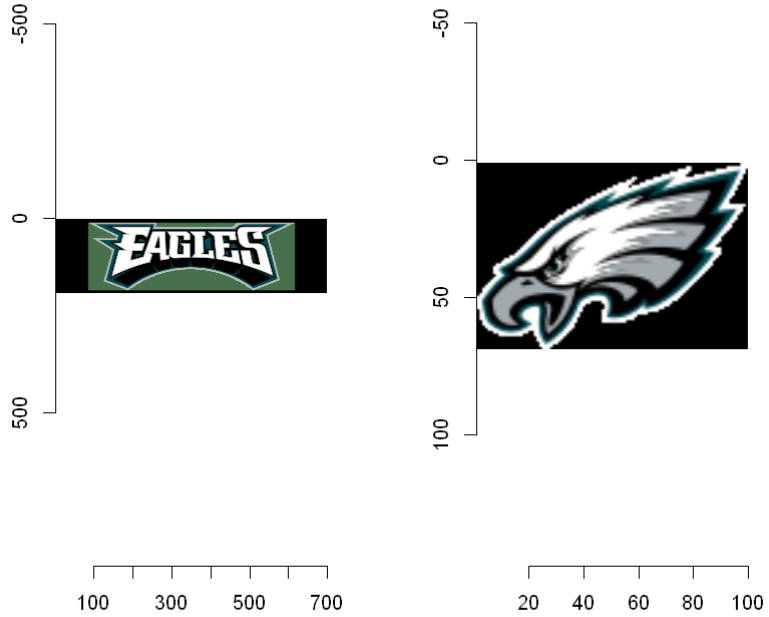

```
[1] "New York Jets >> NYJ"
```



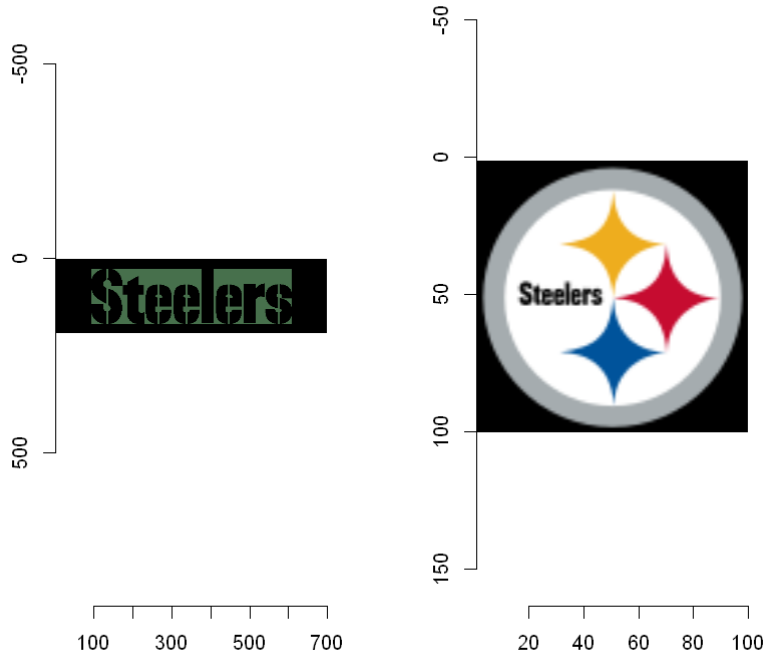

```
[1] "Oakland Raiders >> OAK"
```



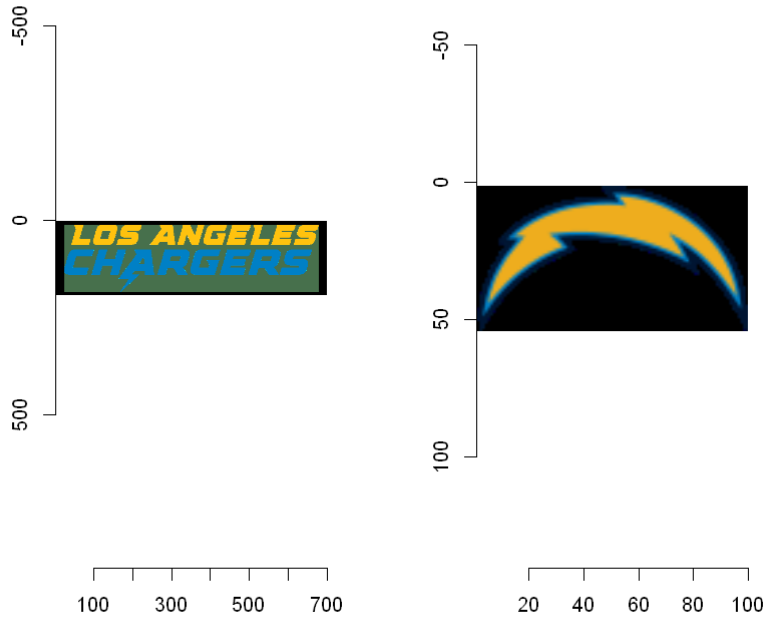

```
[1] "Philadelphia Eagles >> PHI"
```



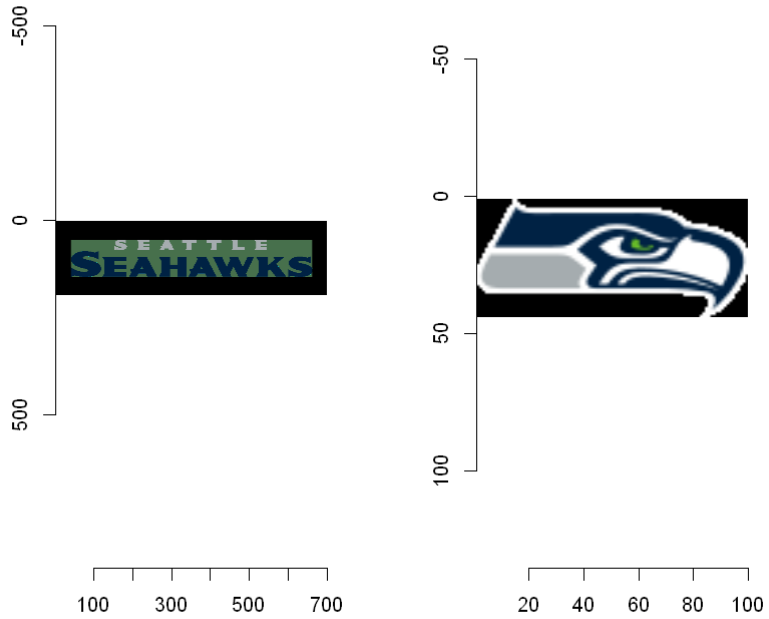

```
[1] "Pittsburgh Steelers >> PIT"
```



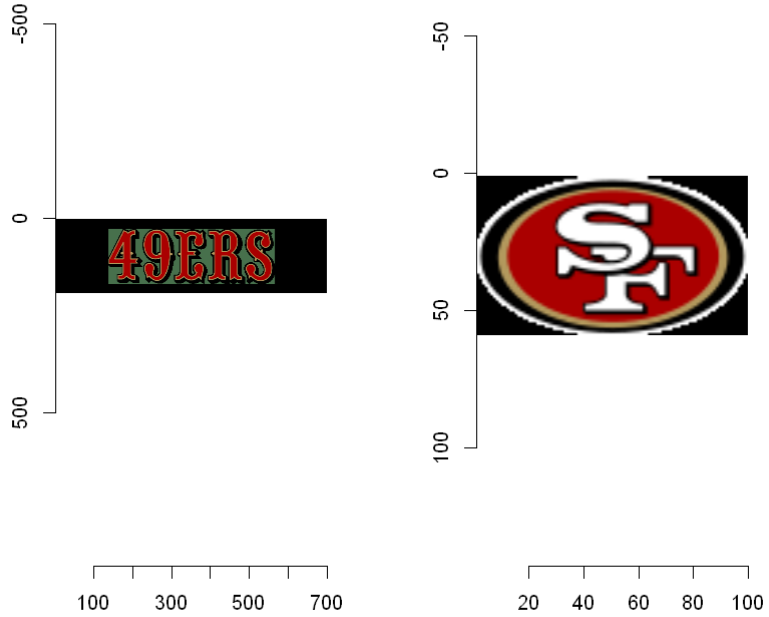

```
[1] "San Diego Chargers >> SD"
```



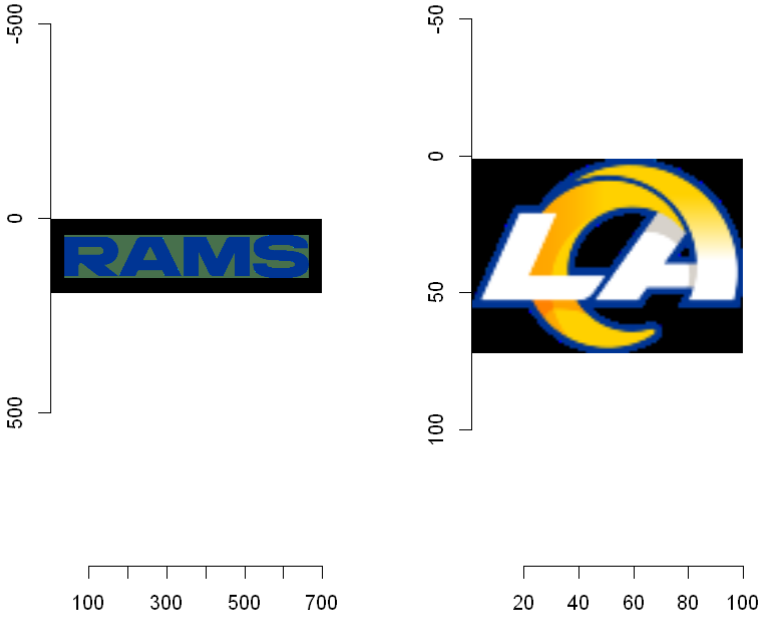

```
[1] "Seattle Seahawks >> SEA"
```



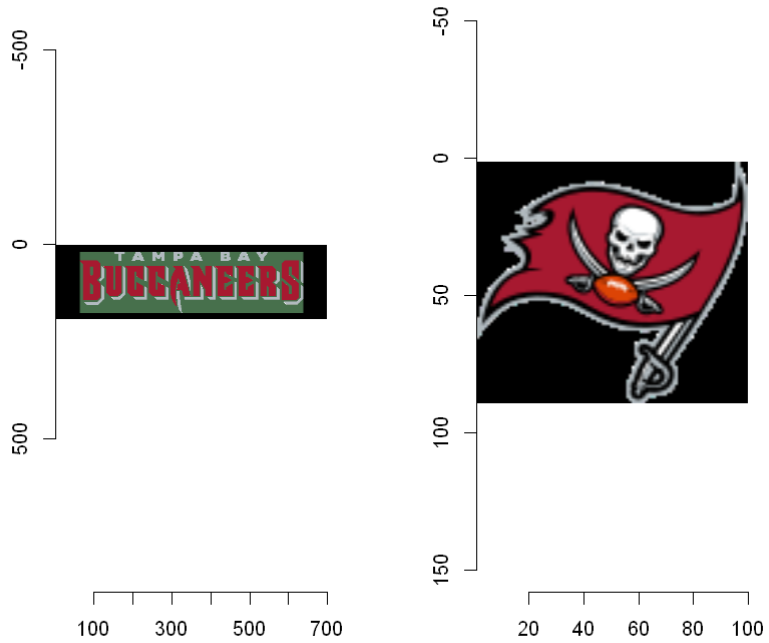
[1] "San Francisco 49ers >> SF"



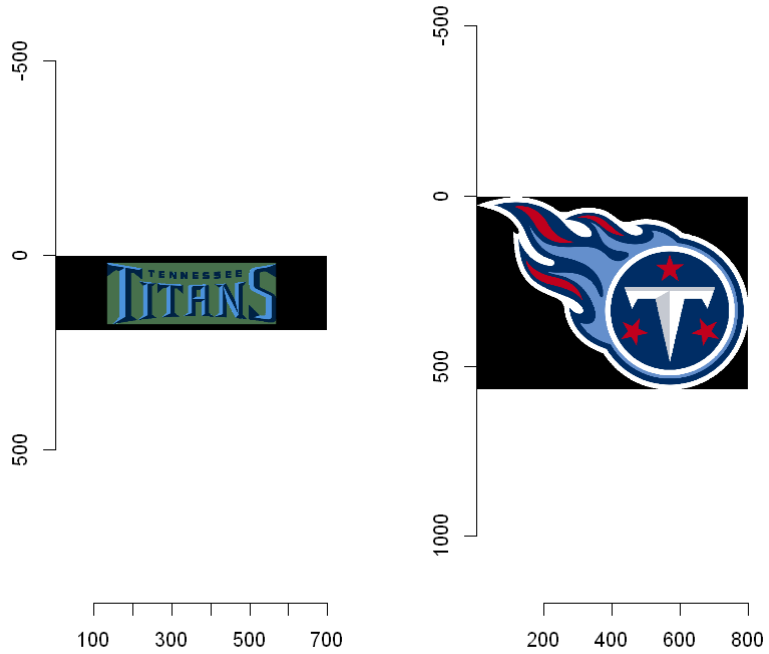

```
[1] "St. Louis Rams >> STL"
```



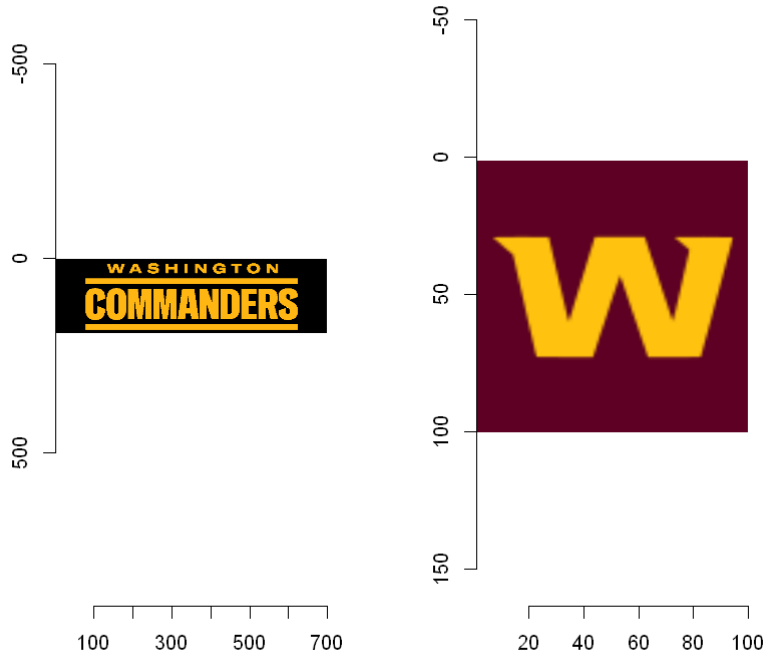

```
[1] "Tampa Bay Buccaneers >> TB"
```




```
[1] "Tennessee Titans >> TEN"
```




```
[1] "Washington Football Team >> WAS"
```



Vejam que é possível ter análises bem complexas e elaboradas, como por exemplo este bloco de código abaixo que foi adaptado do *Exemplo 5* de Get Start with nflfastR.

> Não é objetivo desta disciplina de introdução exigir estes elementos. Coloquei aqui apenas para caráter informativo e mostrar que é possível realizar análises tão complexas quanto desejarmos.

In [ ]:
```
#offense <- temporada %>%
#  dplyr::group_by(posteam) %>%
#  dplyr::summarise(off_epa = mean(epa, na.rm = TRUE))

#defense <- temporada %>%
#  dplyr::group_by(defteam) %>%
#  dplyr::summarise(def_epa = mean(epa, na.rm = TRUE))

#logos <- teams_colors_logos %>% dplyr::select(team_abbr, team_logo_espn)
```

```
#offense %>%
#  dplyr::inner_join(defense, by = c("posteam" = "defteam")) %>%
#  dplyr::inner_join(logos, by = c("posteam" = "team_abbr")) %>%
#  ggplot2::ggplot(aes(x = off_epa, y = def_epa)) +
#  ggplot2::geom_abline(slope = -1.5, intercept = c(.4, .3, .2, .1, 0, -.1, -.2, -.3), alpha = .2) +
#  ggplot2::geom_hline(aes(yintercept = mean(off_epa)), color = "red", linetype = "dashed") +
#  ggplot2::geom_vline(aes(xintercept = mean(def_epa)), color = "red", linetype = "dashed") +
#  ggimage::geom_image(aes(image = team_logo_espn), size = 0.10, asp = 16 / 9) +
#  ggplot2::labs(
#    x = "Ataque EPA/jogada",
#    y = "Defesa EPA/jogada",
#    caption = "Dados: @nflfastR",
#    title = "2014 NFL Ataque e Defesa EPA por jogada"
#  ) +
#  ggplot2::theme_bw() +
#  ggplot2::theme(
#    aspect.ratio = 9 / 16,
#    plot.title = ggplot2::element_text(size = 12, hjust = 0.5, face = "bold")
#  ) +
#  ggplot2::scale_y_reverse()
```

```
Error in loadNamespace(x): there is no package called ‘ggimage’
Traceback:

1. loadNamespace(x)
2. withRestarts(stop(cond), retry_loadNamespace = function() NULL)
3. withOneRestart(expr, restarts[[1L]])
4. doWithOneRestart(return(expr), restart)
```

Manipulação de dados

Criação dos *datasets* segmentados por variáveis

Pense no seguinte problema. Sabendo que o time joga tanto em casa (*home_team*) quanto fora de casa (*away_team*), em qual semana o time escolhido ficou de folga. Ou seja, não há entrada de dados na variável *week*.

Para esta atividade de aprofundamento mantenha o time 'SEA' escolhido, mesmo que você explore outras oportunidades posteriormente.

In [ ]:
```
timeEscolhido <- 'SEA'
```

```r
jogosTimeEscolhido <- temporada %>% filter(home_team == timeEscolhido | away_team == timeEscolhido)

table(jogosTimeEscolhido$away_team, jogosTimeEscolhido$week)
```

```
      1   2   3   5   6   7   8   9  10  11  12  13  14  15  16  17  19  20
ARI   0   0   0   0   0   0   0   0   0   0 154   0   0   0   0   0   0   0
CAR   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 167   0
DAL   0   0   0   0 173   0   0   0   0   0   0   0   0   0   0   0   0   0
DEN   0   0 203   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
GB  172   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 192
LA    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 163   0   0
LV    0   0   0   0   0   0   0 196   0   0   0   0   0   0   0   0   0   0
NE    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
NYG   0   0   0   0   0   0   0   0 172   0   0   0   0   0   0   0   0   0
SEA   0 167   0 177   0 168 157   0   0 167   0 164 182   0 202   0   0   0
SF    0   0   0   0   0   0   0   0   0   0   0   0   0 164   0   0   0   0

     21
ARI   0
CAR   0
DAL   0
DEN   0
GB    0
LA    0
LV    0
NE  192
NYG   0
SEA   0
SF    0
```

Criação dos datasets específicos, segmentando o dataset original, para facilitar a manipulação dos dados e responder às perguntas de negócio.

Utilizando o pacote **Tidyverse**, crie novos conjuntos de dados a partir da função *select*. Garanta que todos datasets estejam fazendo um filtro apenas da semana 1.

> Dica: para o filtro da semana 1, utilize a condição **week==1** na função *filter*

jogo com as variáveis *play_id*, *home_team*, *away_team*, *away_score*, *home_score*, *total*

acoesJogadas com as variáveis *play_id*, *rush_attempt*, pass_attempt, field_goal_attempt, down, time, qtr, ydstogo, yards_gained*

pontuacaoJogadas com as variáveis *play_id*, *posteam*, *defteam*, *posteam_score*, *defteam_score*, *rush*, *pass*, *name*, *passer*, *rusher*, *receiver*, *interception*, *play_type*, *pass_length*, *air_yards*, *kick_distance*, *drive*, *touchdown*, *td_team*

descricaoJogadas com as variáveis *play_id,desc*, *passer_player_name*, *passing_yards*, *receiver_player_name*, *punt_returner_player_name,name*

> Repare que **TODOS** conjuntos de dados criados possuem a variável *play_id*, porque ela fará o relacionamento entre os conjuntos de dados, caso você queria/precise combinar conjuntos de dados para chegar à uma solução

In [ ]:
```
jogo <- jogosTimeEscolhido %>%
    filter(week==1) %>%
    select(play_id,
           home_team, away_team, away_score, home_score, total
          )
```

In [ ]:
```
acoesJogadas <- jogosTimeEscolhido %>%
    filter(week==1) %>%
    select(play_id,
           rush_attempt, pass_attempt, field_goal_attempt, down, time, qtr, ydstogo, yards_gained
          )
```

In [ ]:
```
pontuacaoJogadas <- jogosTimeEscolhido %>%
    filter(week==1) %>%
    select(play_id,
           posteam, defteam, posteam_score, defteam_score, rush, pass, passer, rusher, receiver, interception, play_type, pass_le
          )
```

In [ ]:
```
descricaoJogadas <- jogosTimeEscolhido %>%
    filter(week==1) %>%
    select(play_id,
           desc, passer_player_name, passing_yards, receiver_player_name, punt_returner_player_name
          )
```

In [ ]:
```
head(jogo)
head(acoesJogadas)
head(pontuacaoJogadas)
head(descricaoJogadas)
```

A nflverse_data: 6 × 6

play_id	home_team	away_team	away_score	home_score	total
<dbl>	<chr>	<chr>	<int>	<int>	<int>
1	SEA	GB	16	36	52
36	SEA	GB	16	36	52
58	SEA	GB	16	36	52
79	SEA	GB	16	36	52
111	SEA	GB	16	36	52
132	SEA	GB	16	36	52

A nflverse_data: 6 × 9

play_id	rush_attempt	pass_attempt	field_goal_attempt	down	time	qtr	ydstogo	yards_gained
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>
1	NA	NA	NA	NA	15:00	1	0	NA
36	0	0	0	NA	15:00	1	0	0
58	1	0	0	1	14:56	1	10	6
79	0	0	0	2	14:30	1	4	0
111	1	0	0	1	14:11	1	10	15
132	1	0	0	1	13:32	1	10	2

A nflverse_data: 6 × 19

play_id	posteam	defteam	posteam_score	defteam_score	rush	pass	passer	rusher	receiver	interception	play_type	pass_length	air_yards	kic
<dbl>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<chr>	<chr>	<dbl>	
1	NA	NA	NA	NA	0	0	NA	NA	NA	NA	NA	NA	NA	
36	GB	SEA	0	0	0	0	NA	NA	NA	0	kickoff	NA	NA	
58	GB	SEA	0	0	1	0	NA	E.Lacy	NA	0	run	NA	NA	
79	GB	SEA	0	0	1	0	NA	E.Lacy	NA	0	no_play	NA	NA	
111	GB	SEA	0	0	1	0	NA	E.Lacy	NA	0	run	NA	NA	
132	GB	SEA	0	0	1	0	NA	J.Starks	NA	0	run	NA	NA	

A nflverse_data: 6 × 6

play_id	desc	passer_player_name	passing_yards	receiver_player_name	punt_returner_player_name
<dbl>	<chr>	<chr>	<dbl>	<chr>	<chr>
1	GAME	NA	NA	NA	NA
36	4-S.Hauschka kicks 71 yards from SEA 35 to GB -6. 26-D.Harris to GB 13 for 19 yards (57-M.Morgan).	NA	NA	NA	NA
58	(14:56) 27-E.Lacy right tackle to GB 19 for 6 yards (94-K.Williams).	NA	NA	NA	NA
79	(14:30) 27-E.Lacy left tackle to GB 22 for 3 yards (56-C.Avril, 50-K.Wright). PENALTY on SEA-72-M.Bennett, Defensive Offside, 5 yards, enforced at GB 19 - No Play.	NA	NA	NA	NA
111	(14:11) (Shotgun) 27-E.Lacy up the middle to GB 39 for 15 yards (54-B.Wagner).	NA	NA	NA	NA
132	(13:32) (No Huddle) 44-J.Starks right guard to GB 41 for 2 yards (31-K.Chancellor, 50-K.Wright).	NA	NA	NA	NA

O uso da função *inner_join* no pacote **Tidyverse** é muito útil para combinar conjunto de dados. Veja, nos exemplos abaixo, como fica a combinação destes *datasets* que foram criados anteriormente.

Pense nos seguintes desafios:

1) Combinar o resultado de **pontuacaoJogadas** que tem a informação de quando um time fez *touchdown* (significa que marcou 6 pontos no jogo) e **descricaoJogadas** onde há uma descrição da jogada. Estes conjuntos de dados estão segmentados, cada um deles possui uma parte da informação. Ao combinar estes dois conjuntos de dados é possível ter todas as variáveis juntas como se fossem um único *dataset*. Eles se combinam a partir da variável *play_id*, que é comum entre eles. A partir desta combinação, a manipulação é similar ao que já foi estudado anteriormente.

In [ ]:
```
pontuacaoJogadas %>% #primeiro dataset
inner_join(descricaoJogadas, by='play_id') %>% #segundo dataset combinando com o primeiro
select(play_id, posteam, touchdown, td_team, desc) %>% #seleção de variáveis
filter(touchdown == 1) #filtro de dados
```

A nflverse_data: 6 × 5

play_id	posteam	touchdown	td_team	desc
<dbl>	<chr>	<dbl>	<chr>	<chr>
802	GB	1	GB	(1:30) 30-J.Kuhn left guard for 2 yards, TOUCHDOWN.
1032	SEA	1	SEA	(13:08) (Shotgun) 3-R.Wilson pass short left to 83-R.Lockette for 33 yards, TOUCHDOWN.
1560	SEA	1	SEA	(3:46) (Shotgun) 24-M.Lynch up the middle for 9 yards, TOUCHDOWN.
2965	SEA	1	SEA	(15:00) 24-M.Lynch left tackle for 3 yards, TOUCHDOWN.
3268	GB	1	GB	(9:37) 12-A.Rodgers pass short right to 18-R.Cobb for 3 yards, TOUCHDOWN.
3662	SEA	1	SEA	(2:37) 3-R.Wilson pass short left to 40-D.Coleman for 15 yards, TOUCHDOWN.

2) Mostrar qual foi o jogador do time da casa e quando ele recebeu o primeiro passe que permitiu correr 5 ou mais jardas.

Para isso, é necessário combinar 3 conjuntos de dados. No dataset **jogo** é possível retornar qual é o time da casa. Já em **acoesJogadas** é possível saber quantas jardas foram conquistadas (com a variável *yards_gained*). E por fim, em **pontuacaoJogadas** há o nome de quem correu com a bola (variável *rusher*). Vamos ver como fica essa combinação?

In [ ]:
```
jogo %>% #primeiro dataset
inner_join(acoesJogadas, by='play_id') %>% #segundo dataset combinando com o primeiro
inner_join(pontuacaoJogadas, by='play_id') %>% #terceiro dataset combinando com o primeiro e o segundo
select(play_id, home_team,posteam, rusher, yards_gained, time, qtr ) %>% #seleção de variáveis
filter( posteam == home_team | yards_gained >=5 ) %>% #filtro de dados
head(1) #retorno apenas de 1 linha
```

A nflverse_data: 1 × 7

play_id	home_team	posteam	rusher	yards_gained	time	qtr
<dbl>	<chr>	<chr>	<chr>	<dbl>	<chr>	<dbl>
58	SEA	GB	E.Lacy	6	14:56	1

Desafios de manipulação de dados

Com base no dataset específico **pontuacaoJogadas**, apresente os dados somente quando ouve *rush* ou *pass* na jogada. Garanta que exista também o nome ou abreviatura do time que está atacando (variável *posteam*), além dos nomes dos jogadores que estão fazendo passe, correndo ou recebendo a bola (variáveis *passer*, *rusher* e *receiver*)

In [ ]:
```
# Seu código de resposta vai aqui

pontuacaoJogadas %>% #selecionando o dataset
select( play_id, posteam, rush, pass, passer, rusher, receiver ) %>% #seleção de variáveis
filter( rush == 1 | pass == 1 ) #filtro de dados
```

A nflverse_data: 129 × 7

play_id	posteam	rush	pass	passer	rusher	receiver
<dbl>	<chr>	<dbl>	<dbl>	<chr>	<chr>	<chr>
58	GB	1	0	NA	E.Lacy	NA
79	GB	1	0	NA	E.Lacy	NA
111	GB	1	0	NA	E.Lacy	NA
132	GB	1	0	NA	J.Starks	NA
153	GB	0	1	A.Rodgers	NA	J.Nelson
177	GB	0	1	A.Rodgers	NA	NA
221	SEA	0	1	R.Wilson	NA	P.Harvin
245	SEA	1	0	NA	M.Lynch	NA
266	SEA	0	1	R.Wilson	NA	P.Harvin
290	SEA	0	1	R.Wilson	NA	J.Kearse
314	SEA	1	0	NA	R.Turbin	NA
335	SEA	0	1	R.Wilson	NA	Z.Miller
391	SEA	1	0	NA	P.Harvin	NA
412	SEA	0	1	R.Wilson	NA	NA
435	SEA	1	0	NA	M.Lynch	NA
456	SEA	0	1	R.Wilson	NA	D.Baldwin
541	GB	0	1	A.Rodgers	NA	J.Nelson
565	GB	0	1	A.Rodgers	NA	E.Lacy
589	GB	0	1	A.Rodgers	NA	R.Cobb
611	GB	0	1	A.Rodgers	NA	J.Nelson
666	GB	1	0	NA	E.Lacy	NA
687	GB	1	0	NA	E.Lacy	NA

play_id	posteam	rush	pass	passer	rusher	receiver
<dbl>	<chr>	<dbl>	<dbl>	<chr>	<chr>	<chr>
708	GB	0	1	A.Rodgers	NA	R.Cobb
737	GB	0	1	A.Rodgers	NA	J.Nelson
761	GB	0	1	A.Rodgers	NA	J.Nelson
802	GB	1	0	NA	J.Kuhn	NA
854	SEA	1	0	NA	M.Lynch	NA
886	SEA	1	0	NA	M.Lynch	NA
907	SEA	0	1	R.Wilson	NA	P.Harvin
947	SEA	1	0	NA	M.Lynch	NA
⋮	⋮	⋮	⋮	⋮	⋮	⋮
3024	GB	0	1	A.Rodgers	NA	J.Nelson
3048	GB	0	1	A.Rodgers	NA	E.Lacy
3072	GB	1	0	NA	E.Lacy	NA
3093	GB	0	1	A.Rodgers	NA	R.Cobb
3117	GB	0	1	A.Rodgers	NA	J.Starks
3152	GB	0	1	A.Rodgers	NA	J.Nelson
3176	GB	1	0	NA	J.Starks	NA
3197	GB	0	1	A.Rodgers	NA	J.Nelson
3221	GB	1	0	NA	J.Starks	NA
3247	GB	1	0	NA	J.Starks	NA
3268	GB	0	1	A.Rodgers	NA	R.Cobb
3288	GB	0	1	A.Rodgers	NA	A.Quarless
3328	SEA	0	1	R.Wilson	NA	L.Willson
3352	SEA	1	0	NA	M.Lynch	NA

play_id	posteam	rush	pass	passer	rusher	receiver
<dbl>	<chr>	<dbl>	<dbl>	<chr>	<chr>	<chr>
3373	SEA	0	1	R.Wilson	NA	D.Baldwin
3407	SEA	1	0	NA	M.Lynch	NA
3428	SEA	0	1	R.Wilson	NA	M.Lynch
3452	SEA	1	0	NA	R.Turbin	NA
3473	SEA	1	0	NA	R.Wilson	NA
3494	SEA	0	1	R.Wilson	NA	R.Lockette
3518	SEA	1	0	NA	P.Harvin	NA
3550	SEA	0	1	R.Wilson	NA	NA
3582	SEA	1	0	NA	M.Lynch	NA
3603	SEA	1	0	NA	M.Lynch	NA
3624	SEA	1	0	NA	M.Lynch	NA
3662	SEA	0	1	R.Wilson	NA	D.Coleman
3725	GB	0	1	A.Rodgers	NA	A.Quarless
3747	GB	1	0	NA	D.Harris	NA
3785	GB	0	1	A.Rodgers	NA	A.Quarless
3820	GB	0	1	A.Rodgers	NA	R.Cobb

Utilizando o subconjunto de dados **acoesJogadas** e **pontuacaoJogadas**, crie uma análise que retorne qual foi o jogador que conquistou mais jardas no terceiro quarto.

In [ ]:
```
# Seu código de resposta vai aqui

temp <- acoesJogadas %>% #primeiro dataset
inner_join( pontuacaoJogadas, by='play_id' ) %>% #segundo dataset combinando com o primeiro
select( play_id, posteam, receiver, yards_gained, time, qtr ) %>% #seleção de variáveis
filter( qtr == 3 ) #filtro de dados
final <- temp[complete.cases(temp[4:4]),] %>% #removendo valores "NA" da coluna 4 (yards_gained)
```

```
filter( yards_gained == max( yards_gained ) ) #filtro de dados
final
```

A nflverse_data: 1 × 6

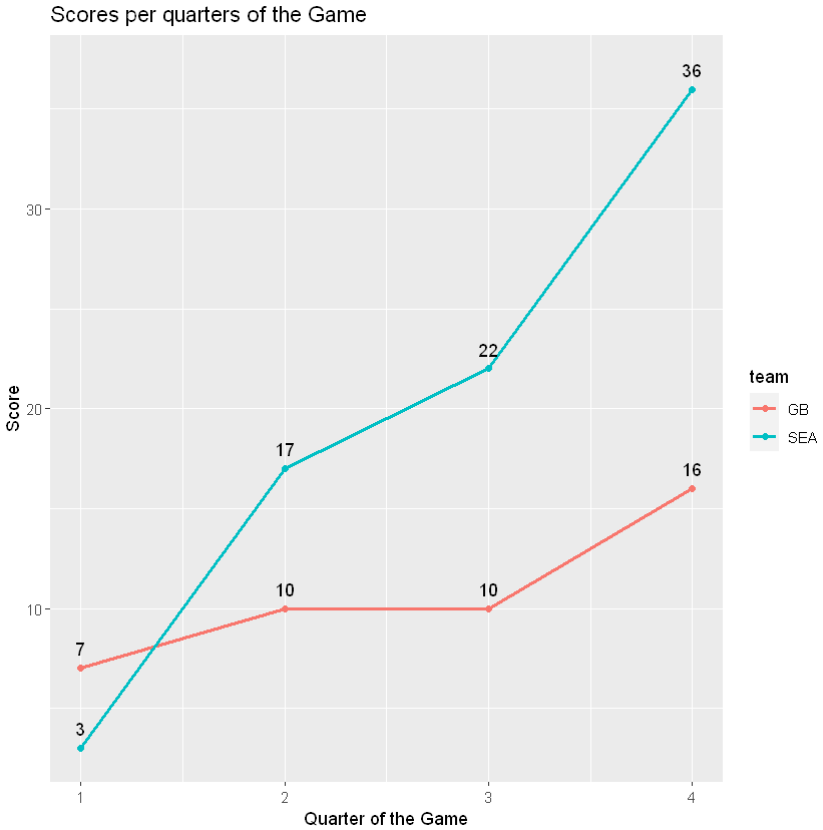
play_id	posteam	receiver	yards_gained	time	qtr
<dbl>	<chr>	<chr>	<dbl>	<chr>	<dbl>
2485	GB	R.Cobb	23	08:17	3

Desafio de geração de gráfico

Crie um gráfico de linhas, mostrando a pontuação de cada time em cada *quarter*. O resultado deve ter duas linhas, uma para cada time, e cada linha será composta pela pontuação de cada um dos *quarters* sendo uma cor para cada time. O eixo X terá os *quarters* e o eixo y terá a pontuação.

In [ ]:
```
# Seu código de resposta vai aqui

# montando o dataset para o gráfico

# preparando os dados
temp <- jogo %>% #primeiro dataset
inner_join( acoesJogadas, by='play_id' ) %>% #segundo dataset combinando com o primeiro
inner_join( pontuacaoJogadas, by='play_id' ) %>% #terceiro dataset combinando com o primeiro e o segundo
select( time, qtr, posteam, defteam, posteam_score, defteam_score ) %>% #seleção de variáveis
filter( time >= "00:00" ) %>% #filtro de dados
unique() #excluindo tuplas duplicadas no dataset

# filtrando os dados do time 1
final1 <- na.omit(temp) %>% #removendo tuplas com valores "NA" do dataset "temp"
group_by( qtr ) %>% #agrupando o dataset "final" pela série "qtr"
filter( qtr %in%  c( 1, 2, 3, 4 ) & time == min( time ) ) %>% #filtro de dados
select( qtr, posteam, posteam_score ) %>% #selecionando novas variáveis
rename( quarter = qtr, team = posteam, score = posteam_score) #renomeando colunas
#final1

# filtrando os dados do time 2
final2 <- na.omit(temp) %>% #removendo tuplas com valores "NA" do dataset "temp"
group_by( qtr ) %>% #agrupando o dataset "final" pela série "qtr"
filter( qtr %in%  c( 1, 2, 3, 4 ) & time == min( time ) ) %>% #filtro de dados
select( qtr, defteam, defteam_score ) %>% #selecionando novas variáveis
```

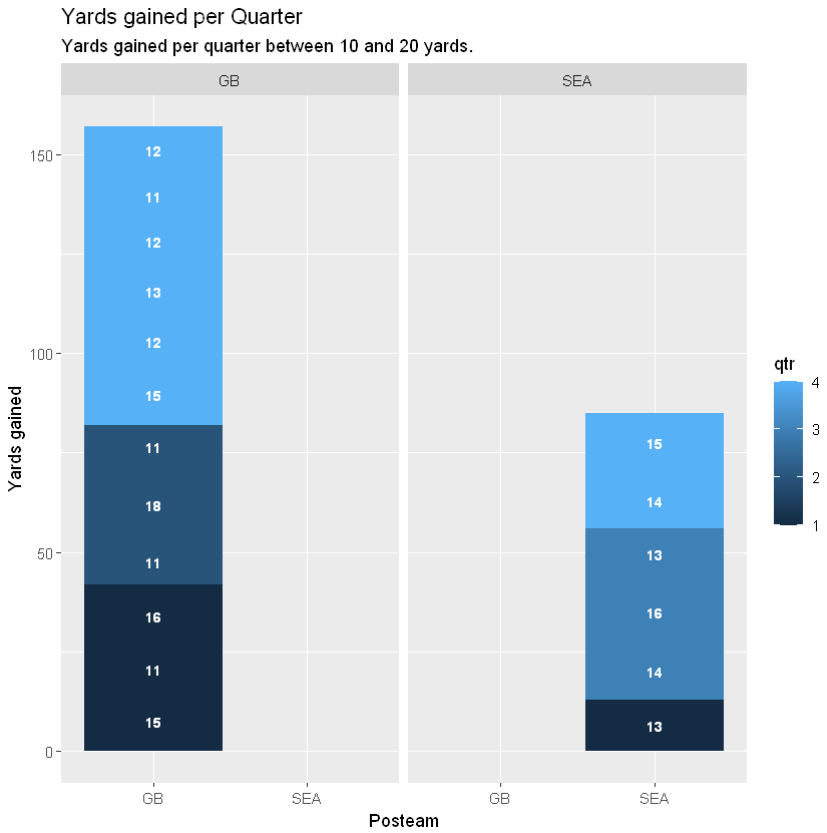
```
rename( quarter = qtr, team = defteam, score = defteam_score) #renomeando colunas
#final2

# unindo os dados dos 2 times em um único dataset
final <- union( final1, final2 )
#final

# plotando o dataset no gráfico
ggplot( final, aes( x = quarter, y = score, color = team, label = score ) ) +
geom_line( size = 1 ) +
geom_point() +
geom_text( nudge_y = 1, color = "black" ) +
xlab("Quarter of the Game") + ylab("Score") +
labs( title = "Scores per quarters of the Game" )
```



Crie um gráfico de barras empilhada (colunas verticais), utilizando somente as jogadas que tiveram entre 10 e 20 jardas conquistadas. O

empilhamento das barras será feito pela quantidade de jardas conquistadas (entre 10 e 20). Mantenha as barras verticais segmentadas por quarter do jogo, e por fim, crie a faceta baseada nos times.

In [ ]:
```
# Seu código de resposta vai aqui

# montando o dataset para o gráfico

# preparando os dados
df <- jogo %>% #primeiro dataset
inner_join( acoesJogadas, by='play_id' ) %>% #segundo dataset combinando com o primeiro
inner_join( pontuacaoJogadas, by='play_id' ) %>% #terceiro dataset combinando com o primeiro e o segundo
select( play_id, posteam, rush_attempt, pass_attempt, field_goal_attempt, qtr, yards_gained ) %>% #seleção de variáveis
filter( yards_gained > 10 & yards_gained < 20 ) #filtro de dados
#df

# plotando o dataset no gráfico
ggplot( df, aes( x = posteam, y = yards_gained, fill = qtr, label = yards_gained ) ) +
geom_bar( stat = "identity" ) +
geom_text( fontface = "bold", size = 3, color = "white", position = position_stack( vjust = 0.5 ) ) +
xlab("Posteam") + ylab("Yards gained") +
labs( title = "Yards gained per Quarter",
      subtitle = "Yards gained per quarter between 10 and 20 yards." ) +
facet_wrap ( ~posteam )
```


Yards gained per Quarter
Yards gained per quarter between 10 and 20 yards.
GB
SEA
150
100
50
0
Yards gained
12
11
12
13
12
15
11
18
11
16
11
15
15
14
13
16
14
13
qtr
4
3
2
1
GB
SEA
GB
SEA
Posteam