679 FP EDA

Ruijian Maggie Lin

2025-04-03

Data Clean

```r
train_data <- read.csv("train_data_wide.csv", na = c("", "NA"))
test_data <- read.csv("test_data_wide.csv", na = c("", "NA"))
```

```r
# 1. Remove rows with ANY missing values in train (complete cases only)
complete_train <- train_data %>%
  drop_na()

# Check if any NAs remain
if(anyNA(complete_train)) {
  cat("Warning: Missing values still exist in the data!\n")

  # Identify which columns still have NAs
  na_columns <- colnames(complete_train)[colSums(is.na(complete_train)) > 0]
  cat("Columns with remaining NAs:", paste(na_columns, collapse = ", "), "\n")
} else {
  cat("Success: No missing values in the cleaned dataset.\n")
}
```

```
## Success: No missing values in the cleaned dataset.
```

```r
# 2. Remove rows with ANY missing values in test (complete cases only)
complete_test <- test_data %>%
  drop_na()

# Check if any NAs remain
if(anyNA(complete_test)) {
  cat("Warning: Missing values still exist in the data!\n")

  # Identify which columns still have NAs
  na_columns <- colnames(complete_test)[colSums(is.na(complete_test)) > 0]
  cat("Columns with remaining NAs:", paste(na_columns, collapse = ", "), "\n")
} else {
  cat("Success: No missing values in the cleaned dataset.\n")
}
```

```
## Success: No missing values in the cleaned dataset.
```

```r
write.csv(complete_train, "train_wide_complete.csv")
write.csv(complete_test, "test_wide_complete.csv")
```

(start run here) Complete Cases CSV Reading

```r
train_data <- read.csv("train_wide_complete.csv")
test_data <- read.csv("test_wide_complete.csv")
```

EDA (using the complete cases)

Participant Count

Raw Dataset

```r
# Final participant count for Train Data
train_final_participant_count <- nrow(train_data)

# Final participant count for Test Data
test_final_participant_count <- nrow(test_data)

cat("Number of participants in train:", train_final_participant_count, "\n")
```

```
## Number of participants in train: 726
```

```r
cat("Number of participants in test:", test_final_participant_count, "\n")
```

```
## Number of participants in test: 306
```

Complete Case (Remove Missingness)

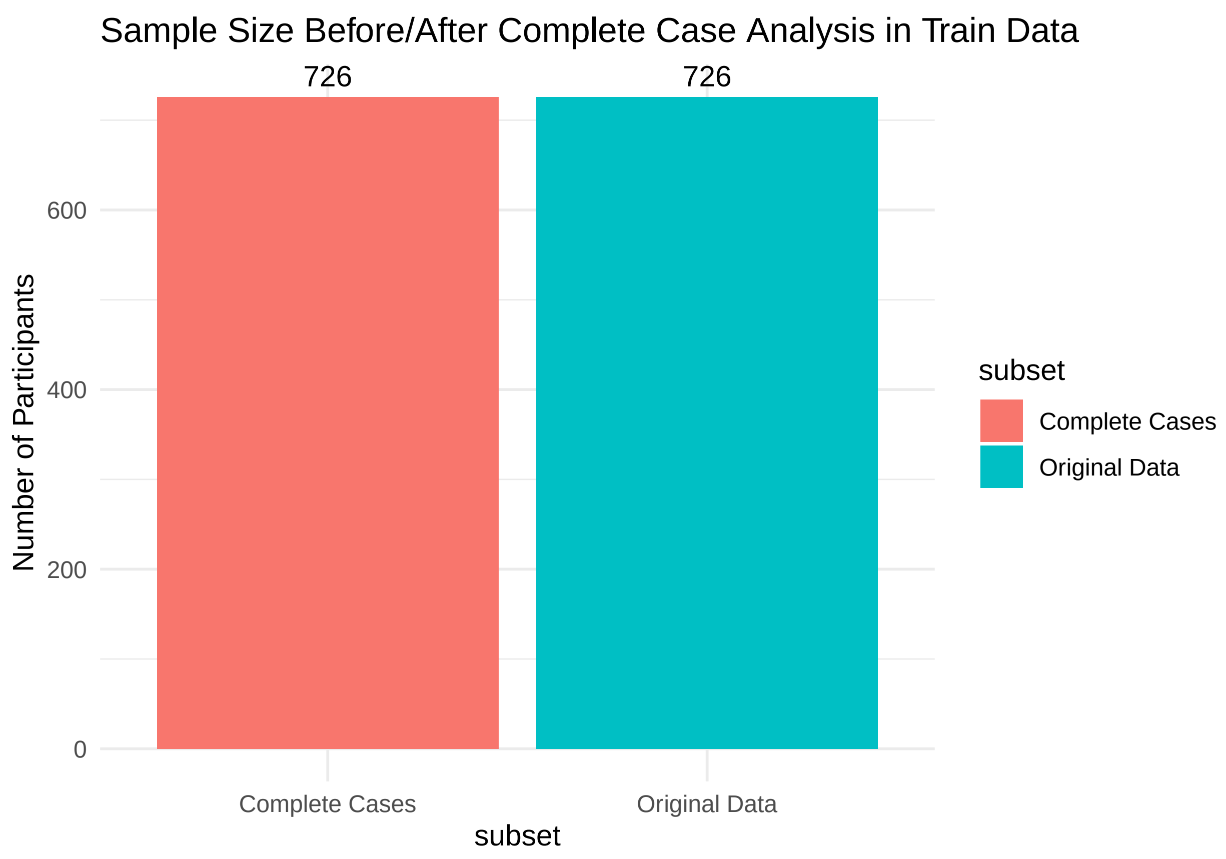
```r
# Final participant count for Complete Train Data
train_final_participant <- nrow(complete_train)

# Final participant count for Complete Test Data
test_final_participant <- nrow(complete_test)

cat("Number of participants in complete train:", train_final_participant, "\n")
```

```
## Number of participants in complete train: 726
```

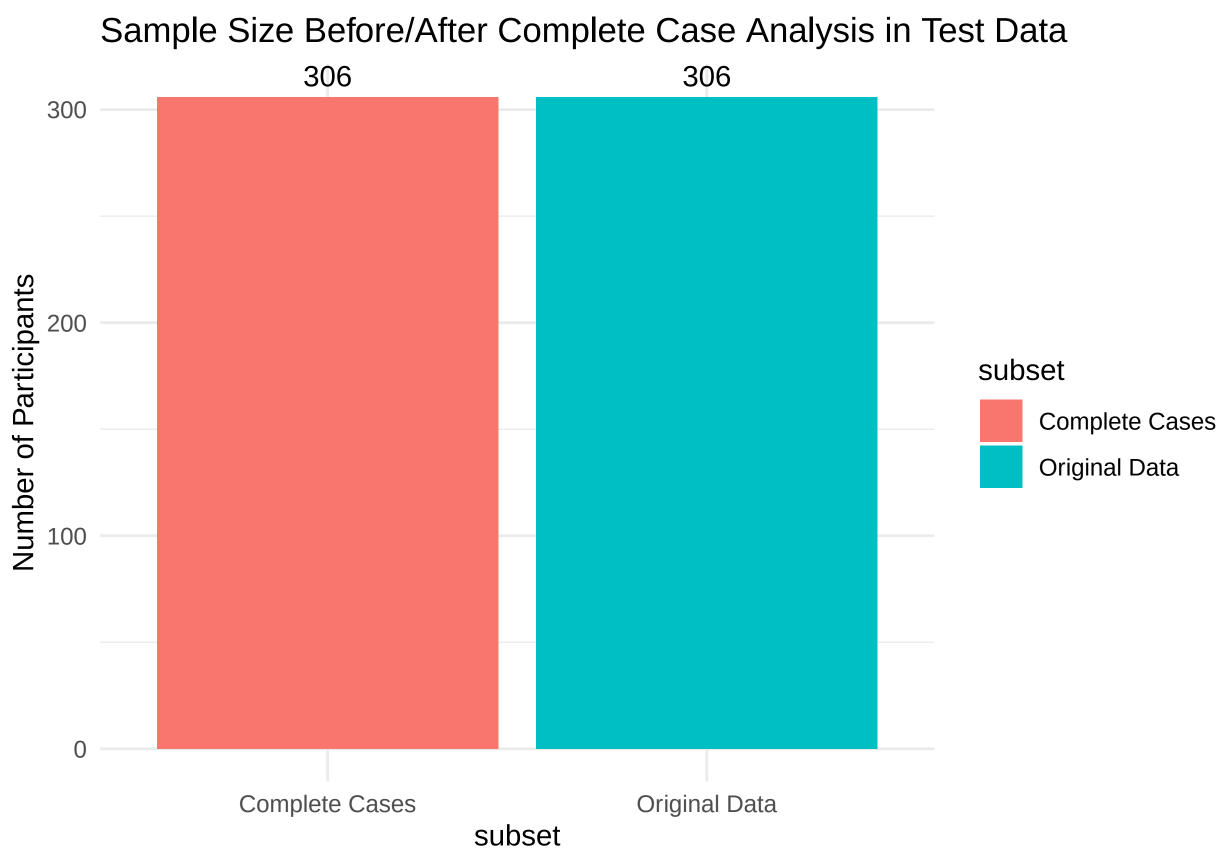
```r
cat("Number of participants in complete test:", test_final_participant, "\n")
```

```
## Number of participants in complete test: 306
```

```r
# Create a function to summarize key variables
summarize_demo <- function(data, label) {
  data %>%
    summarise(
      n = n(),
      .groups = 'drop'
    ) %>%
    mutate(subset = label)
}

# Compare original vs. complete cases in train
demo_comparison_train <- bind_rows(
  summarize_demo(train_data, "Original Data"),
  summarize_demo(complete_train, "Complete Cases")
)

# Compare original vs. complete cases in test
demo_comparison_test <- bind_rows(
  summarize_demo(test_data, "Original Data"),
  summarize_demo(complete_test, "Complete Cases")
)

# Visual comparison in train
ggplot(demo_comparison_train, aes(x = subset, y = n, fill = subset)) +
  geom_col() +
  geom_text(aes(label = n), vjust = -0.5) +
  labs(title = "Sample Size Before/After Complete Case Analysis in Train Data",
       y = "Number of Participants") +
  theme_minimal()
```



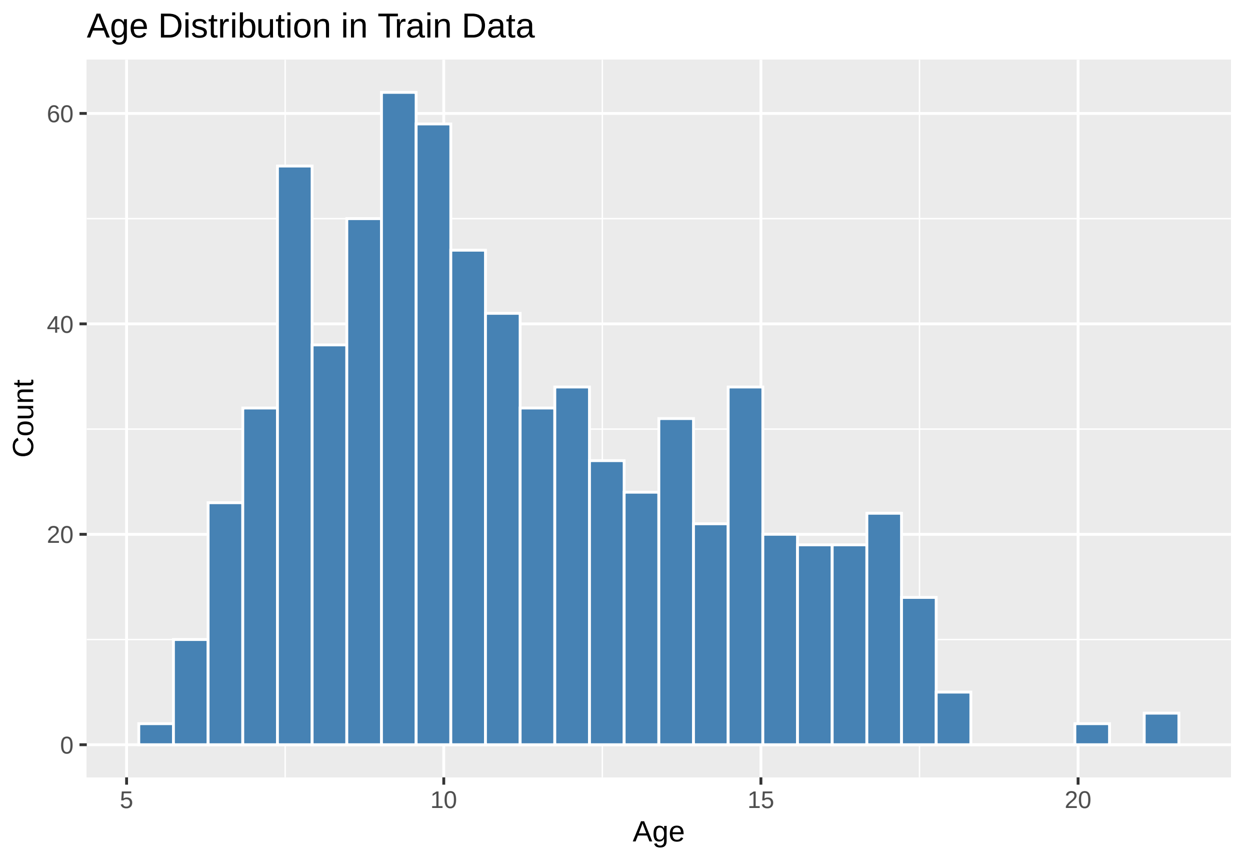
```r
# Visual comparison in test
ggplot(demo_comparison_test, aes(x = subset, y = n, fill = subset)) +
  geom_col() +
  geom_text(aes(label = n), vjust = -0.5) +
  labs(title = "Sample Size Before/After Complete Case Analysis in Test Data",
       y = "Number of Participants") +
  theme_minimal()
```



Demographic & Age Distribution in Train

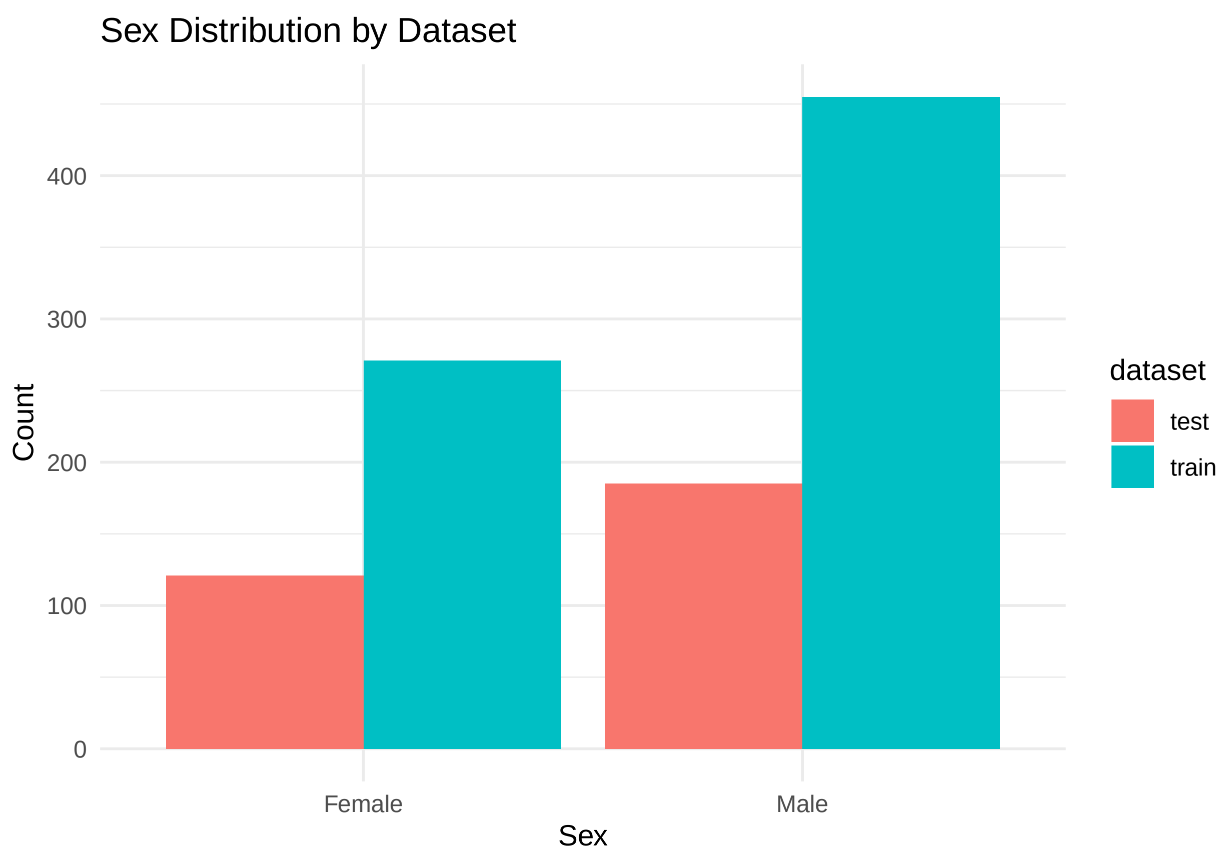
Age Distribution

```r
ggplot(complete_train, aes(x = age)) +
  geom_histogram(bins = 30, fill = "steelblue", color = "white") +
  labs(title = "Age Distribution in Train Data", x = "Age", y = "Count")
```

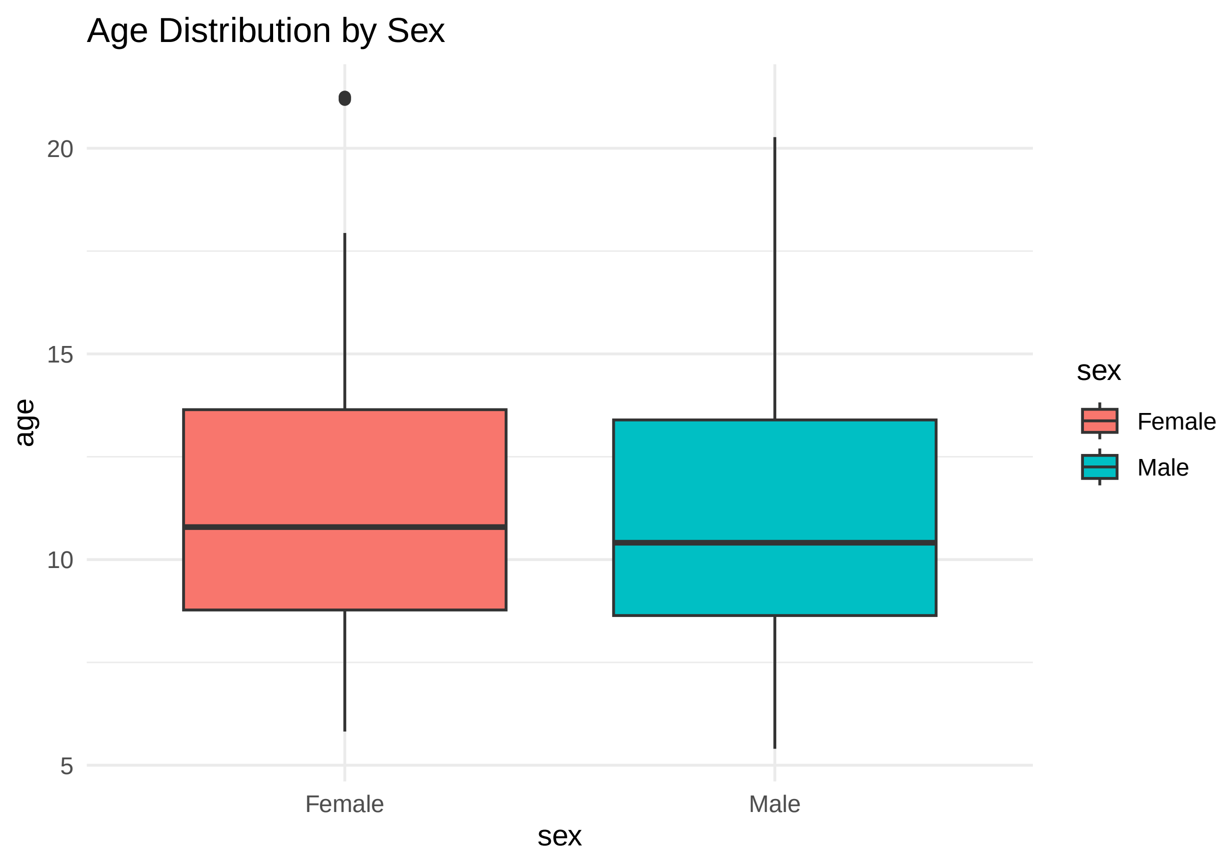


Age and Sex

```r
# Add a column to label the dataset
complete_train <- complete_train %>% mutate(dataset = "train")
complete_test  <- complete_test %>% mutate(dataset = "test")

# Combine both
combined_data <- bind_rows(complete_train, complete_test)

# Plot sex distribution by dataset
ggplot(combined_data, aes(x = sex, fill = dataset)) +
  geom_bar(position = "dodge") +
  labs(title = "Sex Distribution by Dataset", x = "Sex", y = "Count") +
  theme_minimal()
```

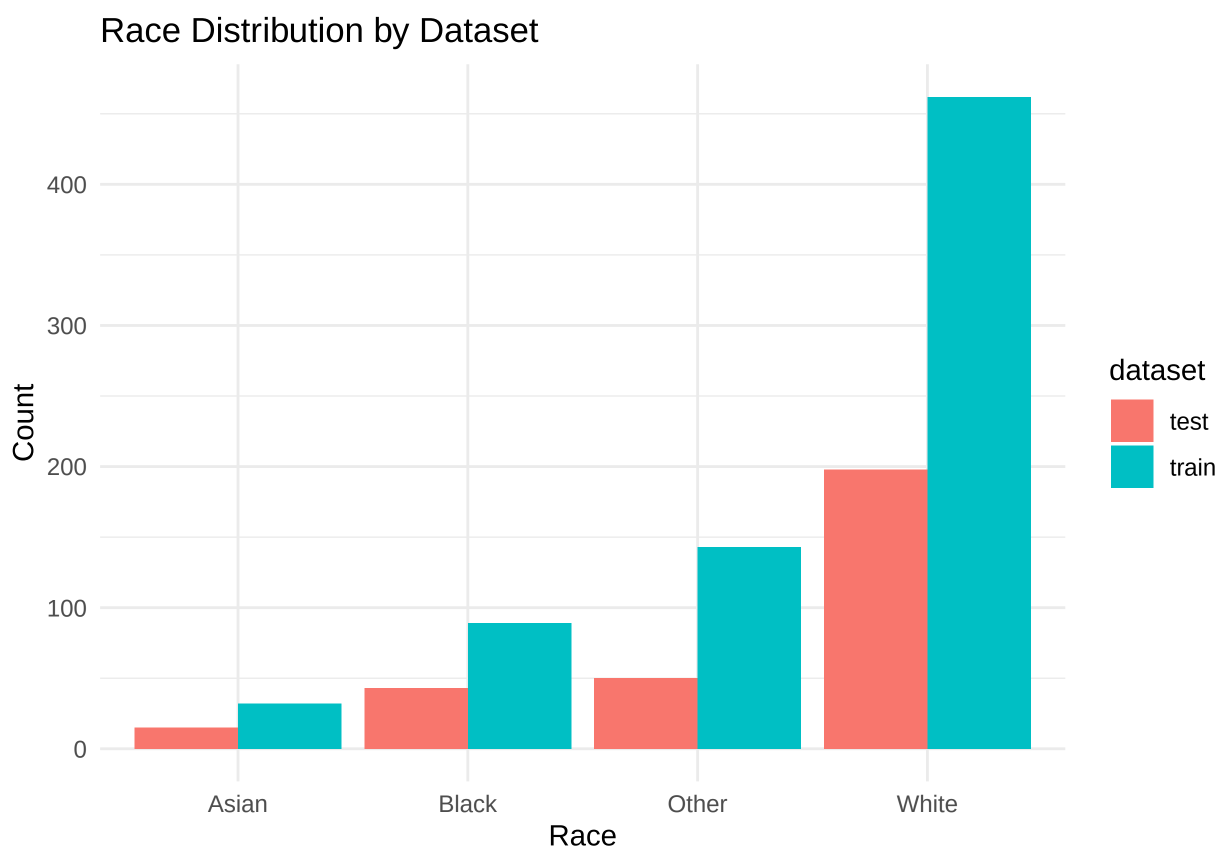


```r
# Age by sex
ggplot(complete_train, aes(x = sex, y = age, fill = sex)) +
  geom_boxplot() +
  labs(title = "Age Distribution by Sex") +
  theme_minimal()
```

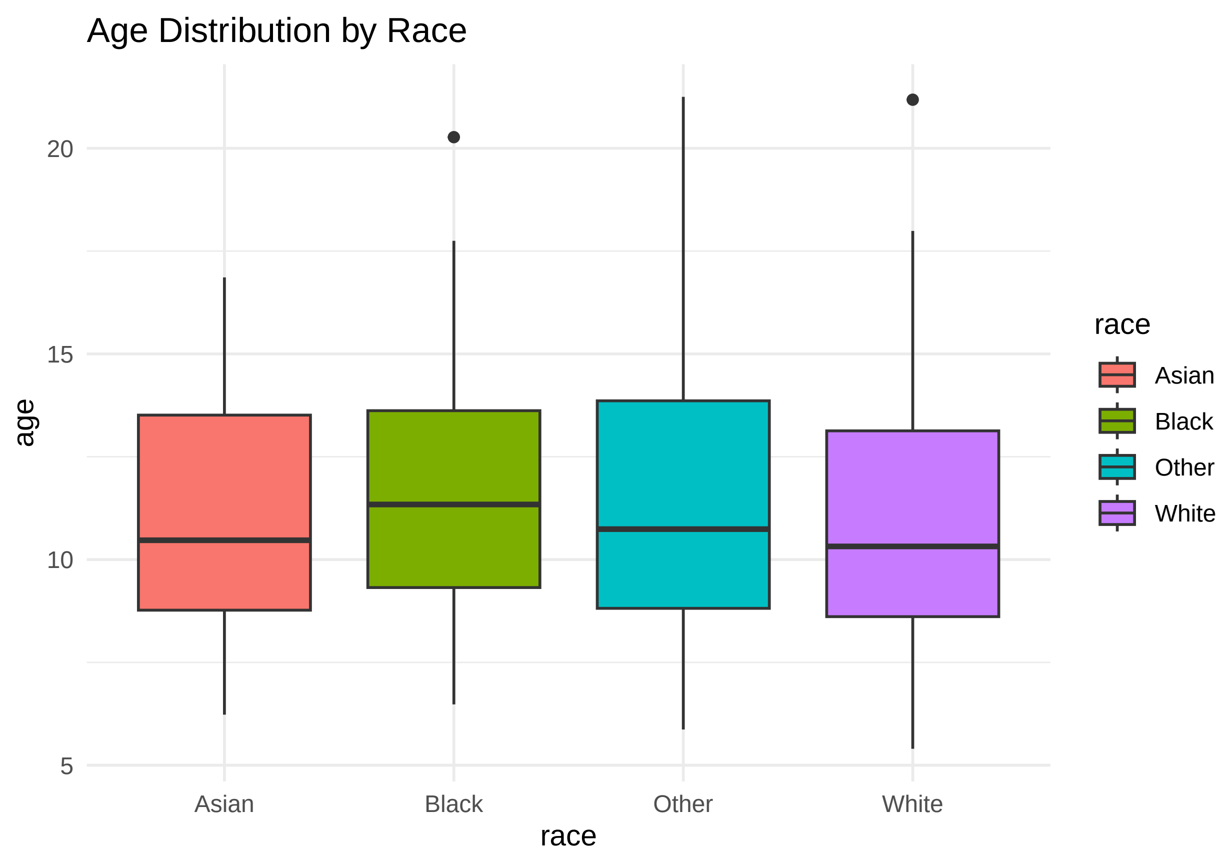


Age and Race

```r
# Plot race distribution by dataset
ggplot(combined_data, aes(x = race, fill = dataset)) +
  geom_bar(position = "dodge") +
  labs(title = "Race Distribution by Dataset", x = "Race", y = "Count") +
  theme_minimal()
```



```r
# Age by race
ggplot(complete_train, aes(x = race, y = age, fill = race)) +
  geom_boxplot() +
  labs(title = "Age Distribution by Race") +
  theme_minimal()
```

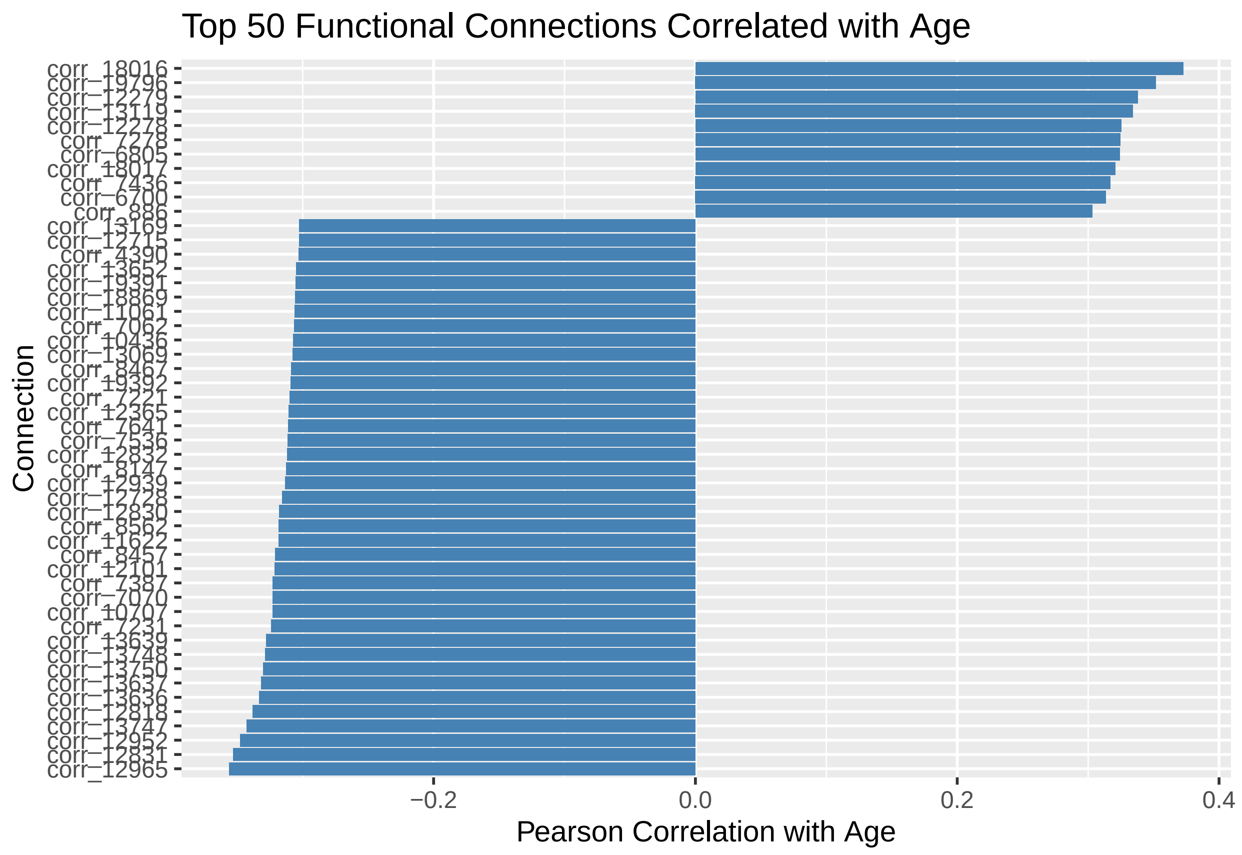


Functional Connectome Matrix Summary

Feature-wise Correlation with Age

Identify which specific connectivities vary most with age.

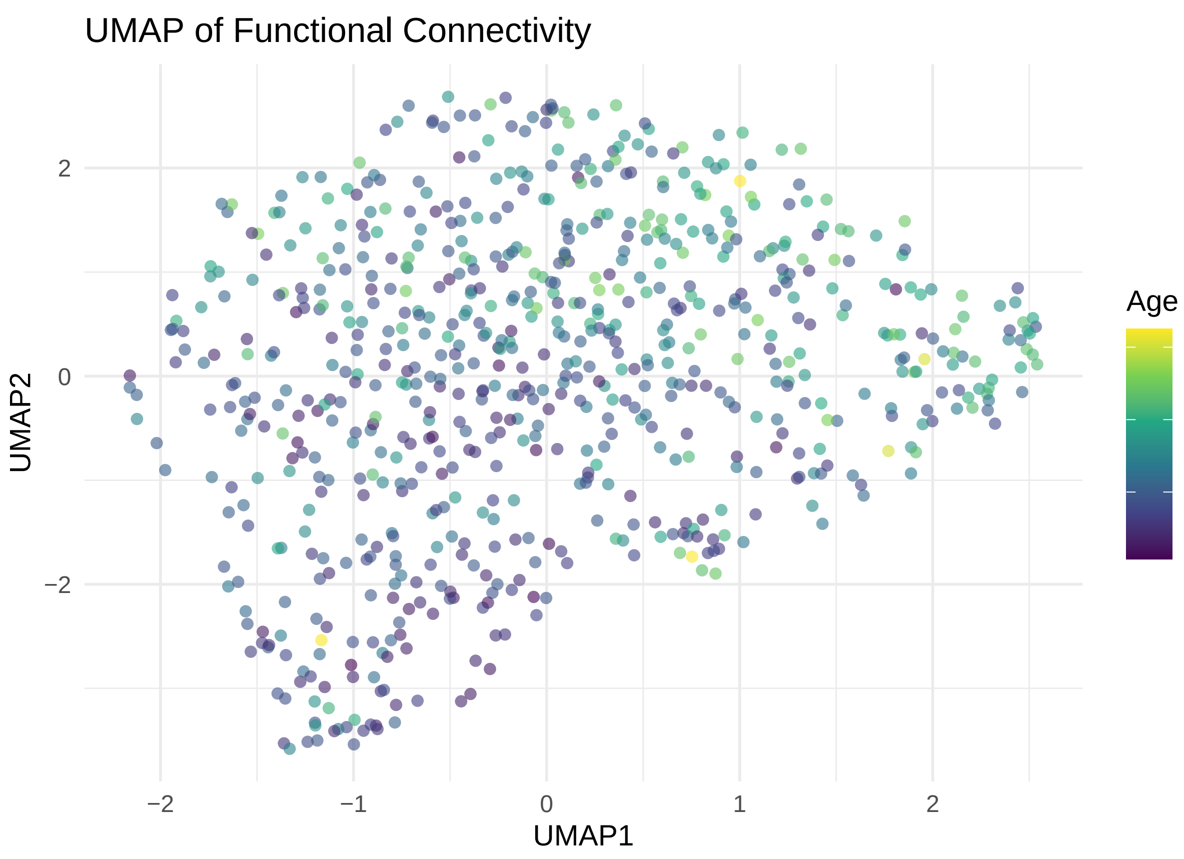
```
cor_results <- data.frame(
  feature = names(complete_train)[grepl("corr_", names(complete_train))],
  correlation = sapply(complete_train[grepl("corr_", names(complete_train))], function(x) cor(x, comple
)

# Sort and plot top correlations
top_corr <- cor_results %>%
  arrange(desc(abs(correlation))) %>%
  slice(1:50)

ggplot(top_corr, aes(x = reorder(feature, correlation), y = correlation)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  labs(title = "Top 50 Functional Connections Correlated with Age",
       x = "Connection", y = "Pearson Correlation with Age")
```



UMAP Colored by Age

Show whether the connectome patterns cluster or change smoothly with age.

```r
connectome_matrix <- complete_train %>% select(starts_with("corr_")) %>% as.matrix()
umap_result <- umap(connectome_matrix)

umap_df <- data.frame(
  UMAP1 = umap_result$layout[,1],
  UMAP2 = umap_result$layout[,2],
  age = complete_train$age
)

ggplot(umap_df, aes(x = UMAP1, y = UMAP2, color = age)) +
  geom_point(alpha = 0.6) +
  scale_color_viridis_c() +
  theme_minimal() +
  labs(title = "UMAP of Functional Connectivity", color = "Age")
```



Each dot = one participant, positioned based on their full-brain functional connectivity pattern (e.g., correlations between ROIs).

UMAP1 and UMAP2 = compressed dimensions capturing global structure and variation in connectivity.

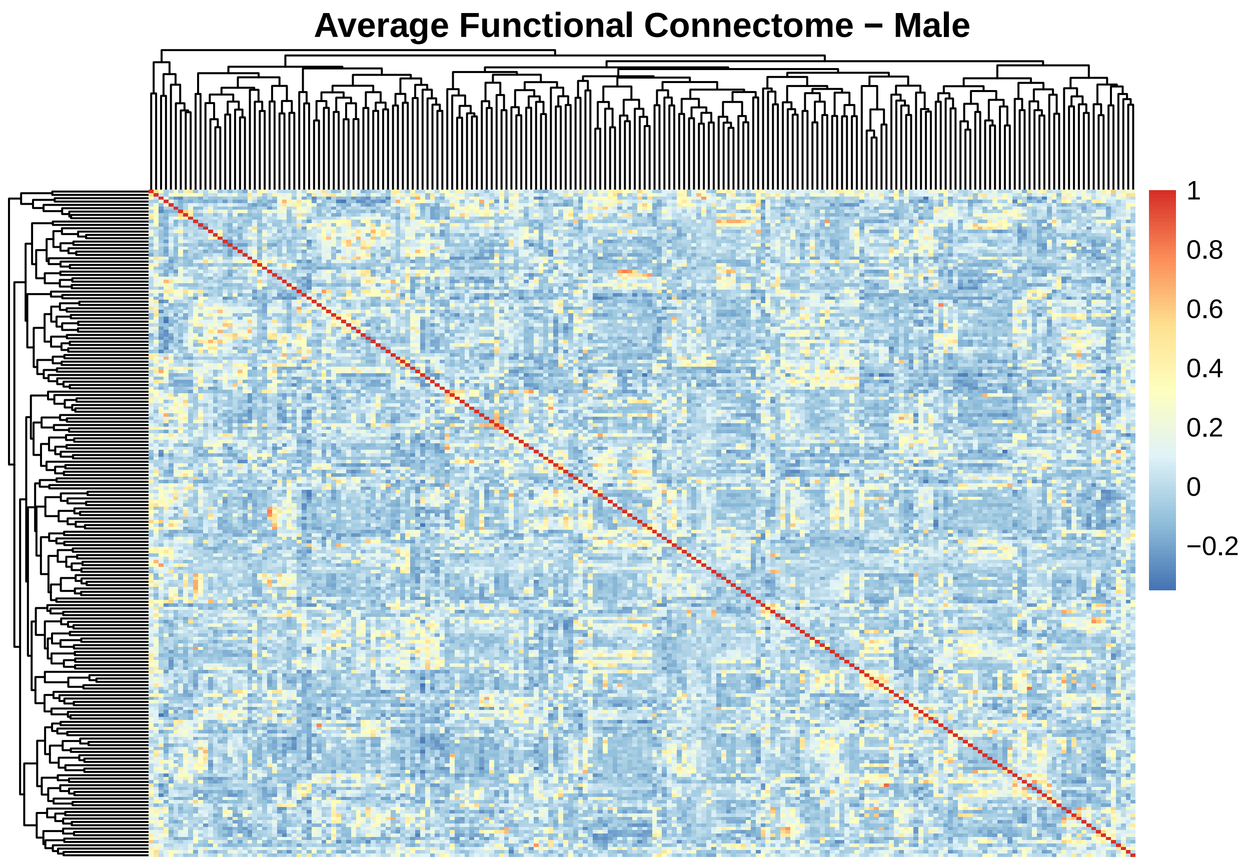
- The **location** of the point is a 2D embedding derived from their 19900 connectome features — UMAP reduces the high-dimensional similarity between participants to 2D.

Interpretation: UMAP projection did not reveal a strong visual gradient in age across the low-dimensional embedding, indicating that age-related variation in functional connectivity may be subtle or non-linear (could be due to high individual variability, or noise in the FC data). Further modeling is needed to quantify the predictive value of connectome features.

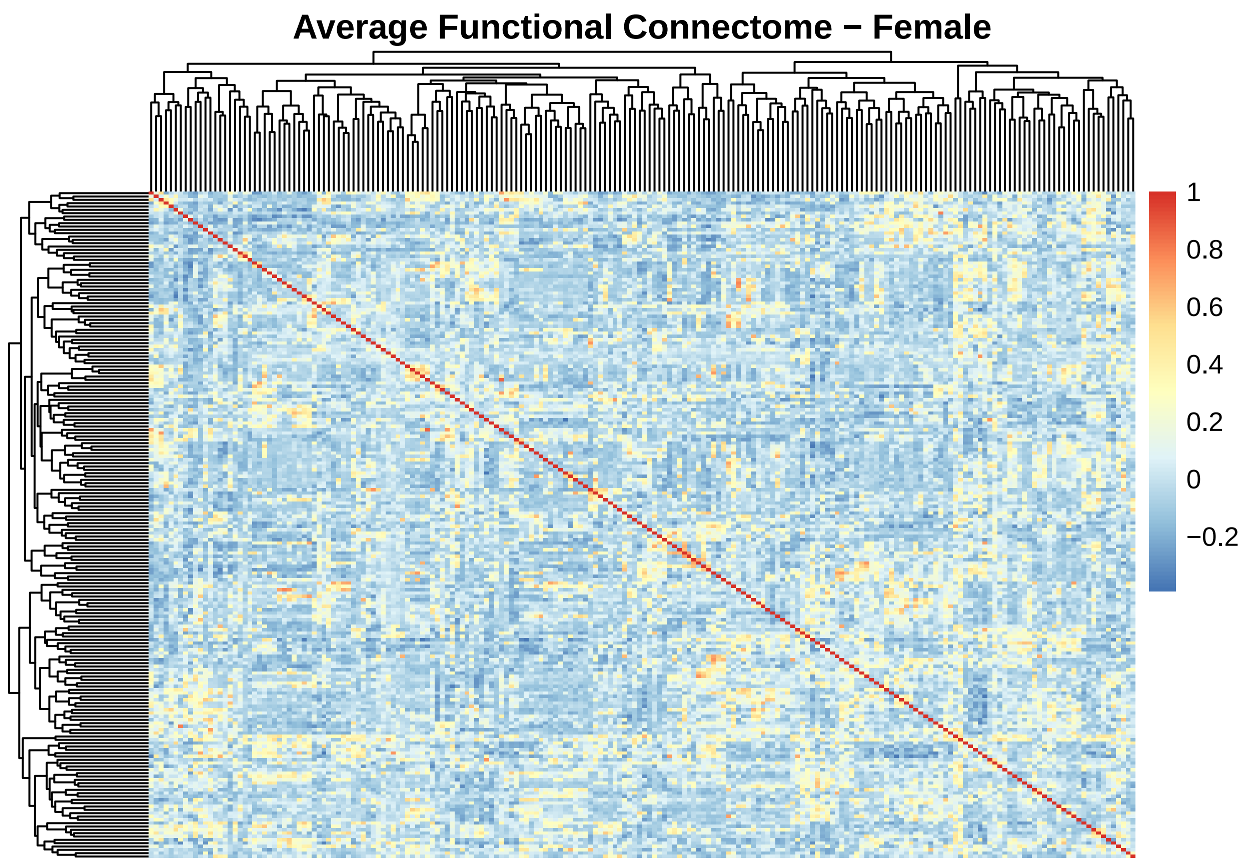
Average Functional Connectome Matrices by Sex

```r
n_regions <- (1 + sqrt(1 + 8 * ncol(connectome_matrix))) / 2

# Indexes by sex
male_idx <- which(complete_train$sex == "Male")
female_idx <- which(complete_train$sex == "Female")

# Average vectors
avg_vec_male <- colMeans(connectome_matrix[male_idx, ])
avg_vec_female <- colMeans(connectome_matrix[female_idx, ])
```

```r
reconstruct_matrix <- function(vec, n) {
  mat <- matrix(0, n, n)
  mat[upper.tri(mat)] <- vec
  mat <- mat + t(mat)  # make symmetric
  diag(mat) <- 1       # optional: set diagonal to 1
  return(mat)
}

# Build full matrices
mat_male <- reconstruct_matrix(avg_vec_male, n_regions)
mat_female <- reconstruct_matrix(avg_vec_female, n_regions)

pheatmap(mat_male, main = "Average Functional Connectome - Male")
```



```r
pheatmap(mat_female, main = "Average Functional Connectome - Female")
```



Interpretation: Visual inspection of the average functional connectome heatmaps reveals more distinct and stronger connectivity blocks in females compared to males. This suggests that females may exhibit stronger or more modular inter-regional synchronization during resting-state fMRI.