Data App Analysis

Prarabdh Bhatia

Introduction

This report evaluates the performance of a pairing algorithm used by the dating app Oz. The algorithm calculates compatibility between users based on their preferences and similarities, assigning scores referred to as Liking Scores and Match Scores. The goal of this analysis is to determine whether the algorithm is accurately identifying the best matches or if its recommendations are flawed. The evaluation focuses on six individuals: three men—Fiyero, Kristoff, and Bruno and three women—Anna, Elphaba, and Rosie. The findings are compared to the algorithm's suggested matches, which pair Anna with Fiyero, Elphaba with Kristoff, and Rosie with Bruno, to check for accuracy.

Methodology

Correlation Matrices: Correlation matrices revealed varied similarities and differences in preferences among users, allowing identification of meaningful comparison users. Men's correlation values were derived from Liked_M_F, while women's values were derived from Liked_F_M.

Liking Scores: The Liking Scores were calculated based on the specified logic and included contributions from both positively and negatively correlated comparison users.

Match Scores: Match Scores were calculated for all nine possible pairs among the first three men and women. The results are summarized in Table 1.

Results The compatibility matrix displays the calculated Match Scores for all relevant pairings.

Based on the calculated Match Scores, the recommended pairings are:

Anna: Kristoff (1.039) Elphaba: Fiyero (0.984) Rosie: Bruno (0.887)

Interpretation

The findings suggest a discrepancy between the Oz algorithm's recommendations and the calculated optimal matches. The highest Match Scores indicate that Anna should be paired with Kristoff, while Elphaba is best matched with Fiyero, and Rosie with Bruno.These pairs differ from the algorithm's initial suggestions, which recommended Anna with Fiyero, Elphaba with Kristoff, and Rosie with Bruno.

This difference in results strongly indicates that the current algorithm is bugged as it fails to prioritize the highest Match Scores based on mutual compatibility. The discrepancies suggest the need for a review and potential recalibration of the Oz algorithm to ensure its outputs align with true compatibility metrics.

```
# Load the data file
load("/Users/prarabdhbhatia/Desktop/Projects/Dating_algorithm_analysis/oz_data.Rdata")

# correlation matrices for male and female preferences
male_corr_matrix <- cor(t(Liked_M_F), use = 'pairwise.complete.obs')
female_corr_matrix <- cor(t(Liked_F_M), use = 'pairwise.complete.obs')

# correlation matrices for male and female preferences
male_corr_matrix <- cor(t(Liked_M_F), use = 'pairwise.complete.obs')
female_corr_matrix <- cor(t(Liked_F_M), use = 'pairwise.complete.obs')
# calculate Liking Score for a target user
```

```r
calc_liking_score <- function(pref_matrix, man_id, woman_id, male_perspective = FALSE) {
# determine matrix and indices
if (male_perspective) {
corr_matrix <- male_corr_matrix
similar_individuals <- which(abs(corr_matrix[, man_id]) > 0.15)
interaction_matrix <- Liked_M_F
} else {
corr_matrix <- female_corr_matrix
similar_individuals <- which(abs(corr_matrix[, woman_id]) > 0.15)
interaction_matrix <- Liked_F_M
}
total_score <- 0
weight_sum <- 0
# iterate over similar individuals
for (index in similar_individuals) {
if (index == ifelse(male_perspective, man_id, woman_id)) {
next
}
# find match potential
shared_interest <- 0
if (!is.na(interaction_matrix[index, ifelse(male_perspective, woman_id, man_id)])) {
if (interaction_matrix[index, ifelse(male_perspective, woman_id, man_id)] == 1 &&
corr_matrix[index, ifelse(male_perspective, man_id, woman_id)] > 0) {
shared_interest <- 1
}
if (interaction_matrix[index, ifelse(male_perspective, woman_id, man_id)] == 0 &&
corr_matrix[index, ifelse(male_perspective, man_id, woman_id)] < 0) {
shared_interest <- 1
}
}
# update scores
total_score <- total_score + abs(corr_matrix[index, ifelse(male_perspective, man_id, woman_id)]) * share
weight_sum <- weight_sum + abs(corr_matrix[index, ifelse(male_perspective, man_id, woman_id)])
}
return(total_score / weight_sum)
}
# function to create compatibility matrix
compatibility_matrix <- matrix(0, nrow = 3, ncol = 3)
for (male_id in seq_len(3)) {
for (female_id in seq_len(3)) {
compatibility_matrix[male_id, female_id] <-
calc_liking_score(Liked_F_M, man_id = male_id, woman_id = female_id, male_perspective = FALSE) + calc_li
}

# full compatibility matrix

full_compatibility_matrix <- outer(seq_len(3), seq_len(3), Vectorize(function(male_id, female_id) { cal

# compatibility scores for the first three men and women

compatibility_matrix_subset <- full_compatibility_matrix[1:3, 1:3]
rownames(compatibility_matrix_subset) <- c("Fiyero", "Kristoff", "Bruno")
colnames(compatibility_matrix_subset) <- c("Anna", "Elphaba", "Rosie")
```

```r
cat("Table 1: Compatibility Matrix for the First Three Men and Women\n")
```

```
## Table 1: Compatibility Matrix for the First Three Men and Women
```

```r
compatibility_matrix_subset
```

```
##               Anna   Elphaba     Rosie
## Fiyero   0.6329698 0.9842220 0.5134620
## Kristoff 1.0399692 0.4601571 0.6040293
## Bruno    0.7679455 0.6327562 0.8870072
```

```r
# recommended matches for each woman based on highest score
recommended_matches <- apply(compatibility_matrix_subset, 2, which.max)
names(recommended_matches) <- c("Anna", "Elphaba", "Rosie")
recommended_matches <- sapply(recommended_matches, function(x) c("Fiyero", "Kristoff", "Bruno")[x])
cat("Recommended Matches for the First Three Women and Men:\n")
```

```
## Recommended Matches for the First Three Women and Men:
```

```r
# recommended matches for each woman based on highest score
recommended_matches <- apply(compatibility_matrix_subset, 2, which.max)
names(recommended_matches) <- c("Anna", "Elphaba", "Rosie")
recommended_matches <- sapply(recommended_matches, function(x) c("Fiyero", "Kristoff", "Bruno")[x])
cat("Recommended Matches for the First Three Women and Men:\n")
```

```
## Recommended Matches for the First Three Women and Men:
```

```r
print(recommended_matches)
```

```
##       Anna    Elphaba      Rosie
## "Kristoff"   "Fiyero"    "Bruno"
```