🎯 Project Title:
“ Data Analysis of Zomato India Restaurant Dataset”

Project Overview – This project analyzes the Zomato India dataset, which contains 9,555 restaurant records and 15 attributes including name, location, cuisines, rating, cost, and service options.
The goal is to explore restaurant trends across cities, popular cuisines, rating patterns, and cost distributions.
Through data cleaning, filtering, grouping, and visualization, we identify insights related to restaurant popularity, affordability, and availability of services like online delivery and table booking.
The analysis aims to support data-driven decisions for customers, restaurant owners, and food app developers.

Objectives

- To analyze the distribution of restaurants across different cities in India.
- To identify popular cuisines and evaluate their average ratings and costs.
- To compare restaurant performance based on online delivery and table booking options.
- To discover high-rated and budget-friendly restaurants using filters and ranking.
- To engineer new features that highlight affordability and popularity for better insights.

Data Analysis Questions -

Level 1: Understanding the Dataset (Basic Exploration)

1. What are the column names and data types in the dataset?
2. Are there any missing values? If yes, in which columns?
3. What is the average rating and cost for two across all restaurants?

Level 2: Data Extraction & Filtering

4. Find the top 10 cities with the most restaurants.
5. Extract restaurants with rating ≥ 4.0 and cost for two under ₹300.
6. Identify restaurants that offer both online delivery and table booking.

Level 3: Grouping & Summarization

7. Compute the average cost and rating for each cuisine type.
8. Determine the most common cuisines across the dataset.
9. Find the city with the highest number of highly rated restaurants.

Level 4: Sorting & Ranking Data

10. Rank restaurants by rating and votes to find the top 5 most popular ones.
11. Find the 5 most affordable restaurants with a rating above 4.
12. Identify the top 5 cities with the most expensive average restaurant cost.

Level 5: Feature Engineering

13. Create a new column PriceCategory:

- Low (cost < ₹300)
- Medium (₹300–₹800)
- High (cost > ₹800)

14. Create a PopularityTag column based on:

- Rating > 4 and Votes > 500 → "Popular"
- Else → "Regular"

- Compute the percentage of popular restaurants in each city.

CODE AND OUTPUT

LEVEL – 1

Use The Desired Libraries

- Library (dplyr)
- Library (ggplot2)

Import The Desired Data Set

```
data <- read.csv("INDIA ZOMATO RESTURAUNT ANALYSIS.csv",
stringsAsFactors = FALSE)
view(INDIA.ZOMATO.RESTURAUNT.ANALYSIS)

# 1. Column names and data types
str(data)
```

```
> str(data)
tibble [8,652 × 22] (S3: tbl_df/tbl/data.frame)
 $ Restaurant.ID       : num [1:8652] 3400025 3400341 3400005 3400021 3400017 ...
 $ Restaurant.Name     : chr [1:8652] "Jahanpanah" "Rangrezz Restaurant" "Time2Eat - Mama Chicken" "Chokho Jeeman Marwari Ja
in Bhojanalya" ...
 $ Country.Code        : num [1:8652] 1 1 1 1 1 1 1 1 1 1 ...
 $ City                : chr [1:8652] "Agra" "Agra" "Agra" "Agra" ...
 $ Address             : chr [1:8652] "E 23, Shopping Arcade, Sadar Bazaar, Agra Cantt, Agra" "E-20, Shopping Arcade, Sadar
Bazaar, Agra Cantt, Agra" "Main Market, Sadar Bazaar, Agra Cantt, Agra" "1/48, Delhi Gate, Station Road, Raja Mandi, Civil L
ines, Agra" ...
 $ Locality            : chr [1:8652] "Agra Cantt" "Agra Cantt" "Agra Cantt" "Civil Lines" ...
 $ Locality.Verbose    : chr [1:8652] "Agra Cantt, Agra" "Agra Cantt, Agra" "Agra Cantt, Agra" "Civil Lines, Agra" ...
 $ Longitude           : num [1:8652] 78 0 78 78 78 ...
 $ Latitude            : num [1:8652] 27.2 0 27.2 27.2 27.2 ...
 $ Cuisines            : chr [1:8652] "North Indian, Mughlai" "North Indian, Mughlai" "North Indian" "Rajasthani" ...
 $ Average.Cost.for.two: num [1:8652] 850 700 500 400 1000 2000 2500 2500 800 3600 ...
 $ Currency            : chr [1:8652] "Indian Rupees(Rs.)" "Indian Rupees(Rs.)" "Indian Rupees(Rs.)" "Indian Rupees(Rs.)"
...
 $ Has.Table.booking   : chr [1:8652] "No" "No" "No" "No" ...
 $ Has.Online.delivery : chr [1:8652] "No" "No" "No" "No" ...
 $ Is.delivering.now   : chr [1:8652] "No" "No" "No" "No" ...
 $ Switch.to.order.menu: chr [1:8652] "No" "No" "No" "No" ...
 $ Price.range         : num [1:8652] 3 2 2 2 3 4 4 4 3 4 ...
 $ Aggregate.rating    : num [1:8652] 3.9 3.5 3.6 4 4.2 4 4.3 4 3.6 3.8 ...
 $ Rating.color        : chr [1:8652] "Yellow" "Yellow" "Yellow" "Green" ...
 $ Rating.text         : chr [1:8652] "Good" "Good" "Good" "Very Good" ...
 $ Votes               : num [1:8652] 140 71 94 87 177 45 133 41 59 46 ...
 $ PriceCategory       : chr [1:8652] "High" "Medium" "Medium" "Medium" ...
>
```

# 2. Check for missing values in each column
colSums(is.na(data))

THERE IS NO ANY MISSING VALUE IN THIS DATA

```
> colSums(is.na(data))
       Restaurant.ID      Restaurant.Name        Country.Code                City             Address
                   0                    0                   0                   0                   0
            Locality     Locality.Verbose           Longitude            Latitude             Cuisines
                   0                    0                   0                   0                   0
Average.Cost.for.two            Currency    Has.Table.booking  Has.Online.delivery    Is.delivering.now
                   0                    0                   0                   0                   0
Switch.to.order.menu         Price.range    Aggregate.rating        Rating.color         Rating.text
                   0                    0                   0                   0                   0
               Votes       PriceCategory
                   0                    0
```

# 3. Average rating and average cost for two
avg_rating <- mean(data$Aggregate.rating, na.rm = TRUE)
avg_cost <- mean(data$Average.Cost.for.two, na.rm = TRUE)

cat("Average Rating:", avg_rating, "\n")
cat("Average Cost for Two:", avg_cost, "\n")

```
                   0                    0
> avg_rating <- mean(data$Aggregate.rating, na.rm = TRUE)
> avg_cost <- mean(data$Average.Cost.for.two, na.rm = TRUE)
> cat("Average Rating:", avg_rating, "\n")
Average Rating: 2.523324
> cat("Average Cost for Two:", avg_cost, "\n")
Average Cost for Two: 623.3703
```

LEVEL – 2

```
# 4. Top 10 cities with the most restaurants
top_cities <- data %>%
  count(City, sort = TRUE) %>%
  head(10)
print(top_cities)
```

```
> top_cities <- data %>%
+   count(City, sort = TRUE) %>%
+   head(10)
> print(top_cities)
# A tibble: 10 × 2
   City            n
   <chr>       <int>
 1 New Delhi    5473
 2 Gurgaon      1118
 3 Noida        1080
 4 Faridabad     251
 5 Ghaziabad      25
 6 Ahmedabad      21
 7 Amritsar       21
 8 Bhubaneshwar   21
 9 Guwahati       21
10 Lucknow        21
```

```
# 5. Restaurants with rating ≥ 4.0 and cost for two under ₹300
high_rating_low_cost <- data %>%
  filter(Aggregate.rating >= 4.0, Average.Cost.for.two < 300)
print(high_rating_low_cost)
```

```
> high_rating_low_cost <- data %>%
+   filter(Aggregate.rating >= 4.0, Average.Cost.for.two < 300)
> print(high_rating_low_cost)
# A tibble: 45 x 22
   Restaurant.ID Restaurant.Name  Country.Code City  Address Locality Locality.Verbose Longitude Latitude Cuisines Average.Cost.for.two Currency Has.Table.booking
           <dbl> <chr>                   <dbl> <chr> <chr>   <chr>    <chr>                <dbl>    <dbl> <chr>                 <dbl> <chr>    <chr>
 1      3400346 Sheroes Hangout             1 Agra  Opposi… Tajganj  Tajganj, Agra         78.0     27.2 Cafe, N…                0 Indian … No
 2     18204507 Ahuja Milk Bhan…            1 Amri… Dhab K… Hathi G… Hathi Gate, Amr…      0        0   Beverag…               100 Indian … No
 3      2200175 Gurdas Ram Jale…            1 Amri… Near G… Town Ha… Town Hall, Amri…      0        0   Mithai                 100 Indian … No
 4      2200153 Kanha Sweets                1 Amri… Shop 1… White A… White Avenue, A…      0        0   Fast Fo…               150 Indian … No
 5      2600109 Sagar Gaire Fas…            1 Bhop… 10, Nu… Arera C… Arera Colony, B…     77.4     23.2 Fast Fo…               250 Indian … No
 6     18279289 BMG - All Day D…            1 Dehr… 140 A,… Jakhan   Jakhan, Dehradun     78.1     30.4 Chinese…                0 Indian … No
 7     18254231 Oye24                       1 Indo… HIG-LI… HIG-LIG  HIG-LIG, Indore      75.9     22.8 North I…               250 Indian … No
 8       801675 Frozen Factory              1 Luck… 1/99, … Gomti N… Gomti Nagar, Lu…      0        0   Dessert…               200 Indian … No
 9       300178 Chaina Ram Sind…            1 New … 6499, … Chandni… Chandni Chowk, …     77.2     28.7 Mithai,…               200 Indian … No
10         1155 Giani's di Hatti            1 New … 651/52… Chandni… Chandni Chowk, …     77.2     28.7 Ice Cre…               150 Indian … No
```

6. Restaurants that offer both online delivery and table booking

```
online_table <- data %>%
  filter(Has.Table.booking == "Yes", Has.Online.delivery == "Yes")
print(online_table)
```

```
> online_table <- data %>%
+   filter(Has.Table.booking == "Yes", Has.Online.delivery == "Yes")
> print(online_table)
# A tibble: 427 x 22
   Restaurant.ID Restaurant.Name  Country.Code City  Address Locality Locality.Verbose Longitude Latitude Cuisines Average.Cost.for.two Currency Has.Table.booking
           <dbl> <chr>                   <dbl> <chr> <chr>   <chr>    <chr>                <dbl>    <dbl> <chr>                 <dbl> <chr>    <chr>
 1        50943 Sultans of Spice             1 Bang… BluPet… BluPeta… BluPetal Hotel,…      77.6     12.9 North I…              1300 Indian … Yes
 2        58268 The Fatty Bao -…             1 Bang… 610, 3… Indiran… Indiranagar, Ba…      77.6     13.0 Asian                 2400 Indian … Yes
 3        58882 Big Brewsky                  1 Bang… Behind… Sarjapu… Sarjapur Road, …      77.7     12.9 Finger …              1800 Indian … Yes
 4        69024 That Madras Pla…             1 Chen… 34/29,… Adyar    Adyar, Chennai        80.3     13.0 Europea…               800 Indian … Yes
 5        72475 Haunted                      1 Chen… 273, F… Anna Na… Anna Nagar East…      80.2     13.1 North I…               800 Indian … Yes
 6        70431 Pantry d'or                  1 Chen… 21/11,… Anna Na… Anna Nagar East…      80.2     13.1 Contine…               750 Indian … Yes
 7        71443 Palmshore                    1 Chen… 95, Ja… Ashok N… Ashok Nagar, Ch…      80.2     13.0 North I…               850 Indian … Yes
 8        73088 Chili's                      1 Chen… 49 & 5… Express… Express Avenue …      80.3     13.1 Mexican…              1700 Indian … Yes
 9        70092 Kaidi Kitchen                1 Chen… 20/3, … Mylapore Mylapore, Chenn…      80.3     13.0 Italian…              1500 Indian … Yes
10        70393 Bombay Brasserie             1 Chen… 3, Col… Nungamb… Nungambakkam, C…      80.3     13.1 North I…              1200 Indian … Yes
```

LEVEL – 3

7. Compute the average cost and rating for each cuisine type

```
# Split multiple cuisines into separate rows
cuisine_data <- data %>%  separate_rows(Cuisines, sep = ",\\s*") %>%
  group_by(Cuisines) %>% summarise(
    Avg_Cost = mean(Average.Cost.for.two, na.rm = TRUE),
    Avg_Rating = mean(Aggregate.rating, na.rm = TRUE),
    Count = n()
  ) %>%  arrange(desc(Count))
print(cuisine_data)
```

```
> cuisine_data <- data %>%
+   separate_rows(Cuisines, sep = ",\\s*") %>%
+   group_by(Cuisines) %>%
+   summarise(
+     Avg_Cost = mean(Average.Cost.for.two, na.rm = TRUE),
+     Avg_Rating = mean(Aggregate.rating, na.rm = TRUE),
+     Count = n()
+   ) %>%
+   arrange(desc(Count))
> print(cuisine_data)
# A tibble: 90 × 4
   Cuisines      Avg_Cost Avg_Rating Count
   <chr>            <dbl>      <dbl> <int>
 1 North Indian      691.       2.51  3946
 2 Chinese           699.       2.60  2690
 3 Fast Food         437.       2.55  1963
 4 Mughlai           728.       2.61   992
 5 Bakery            398.       2.40   726
 6 Continental      1271.       3.51   724
 7 Italian          1189.       3.50   682
 8 South Indian      480.       2.46   631
 9 Cafe              692.       3.23   627
10 Desserts          430.       2.86   597
```

```
# 8. Most common cuisines across the dataset
most_common_cuisines <- cuisine_data %>%
  arrange(desc(Count)) %>%
  head(10)
print(most_common_cuisines)
```

```
> most_common_cuisines <- cuisine_data %>%
+   arrange(desc(Count)) %>%
+   head(5)
> print(most_common_cuisines)
# A tibble: 5 × 4
  Cuisines     Avg_Cost Avg_Rating Count
  <chr>           <dbl>      <dbl> <int>
1 North Indian     691.       2.51  3946
2 Chinese          699.       2.60  2690
3 Fast Food        437.       2.55  1963
4 Mughlai          728.       2.61   992
5 Bakery           398.       2.40   726
```

```
# 9. City with the highest number of highly rated restaurants (rating >= 4)
top_city_high_rated <- data %>%
  filter(Aggregate.rating >= 4.0) %>%
  count(City, sort = TRUE) %>%
  slice(1,2,3)
```

```
> top_city_high_rated <- data %>%
+   filter(Aggregate.rating >= 4.0) %>%
+   count(City, sort = TRUE) %>%
+   slice(1,2,3)
> print(top_city_high_rated)
# A tibble: 3 × 2
  City          n
  <chr>     <int>
1 New Delhi   328
2 Gurgaon      95
3 Noida        29
```

LEVEL – 4

```
# 10. Top 5 most popular restaurants by rating and votes
top_popular <- data %>%
  arrange(desc(Aggregate.rating), desc(Votes)) %>%
  select(Restaurant.Name, City, Aggregate.rating, Votes) %>%
  head(5)
print(top_popular)
```

```
> top_popular <- data %>%
+   arrange(desc(Aggregate.rating), desc(Votes)) %>%
+   select(Restaurant.Name, City, Aggregate.rating, Votes) %>%
+   head(5)
> print(top_popular)
# A tibble: 5 × 4
  Restaurant.Name           City      Aggregate.rating Votes
  <chr>                     <chr>                <dbl> <dbl>
1 Barbeque Nation           Kolkata                4.9  5966
2 AB's - Absolute Barbecues Hyderabad              4.9  5434
3 Mirchi And Mime           Mumbai                 4.9  3244
4 Naturals Ice Cream        New Delhi              4.9  2620
5 Indian Accent - The Manor New Delhi              4.9  1934
```

```
# 11. 5 most affordable restaurants with a rating above 4
affordable_high_rating <- data %>%
  filter(Aggregate.rating > 4.0) %>%
  arrange(Average.Cost.for.two) %>%
  select(Restaurant.Name, City, Average.Cost.for.two, Aggregate.rating)
%>%
  head(5)

print(affordable_high_rating)
```

```
> affordable_high_rating <- data %>%
+   filter(Aggregate.rating > 4.0) %>%
+   arrange(Average.Cost.for.two) %>%
+   select(Restaurant.Name, City, Average.Cost.for.two, Aggregate.rating) %>%
+   head(5)
> print(affordable_high_rating)
# A tibble: 5 × 4
  Restaurant.Name           City      Average.Cost.for.two Aggregate.rating
  <chr>                     <chr>                    <dbl>            <dbl>
1 Sheroes Hangout           Agra                         0              4.9
2 BMG - All Day Dining      Dehradun                     0              4.3
3 Jung Bahadur Kachori Wala New Delhi                   50              4.1
4 Gurdas Ram Jalebi Wala    Amritsar                   100              4.1
5 Ashok Chaat Corner        New Delhi                  100              4.1
```

12. Top 5 cities with the most expensive average restaurant cost

```
expensive_cities <- data %>%
  group_by(City) %>%
  summarise(Avg_Cost = mean(Average.Cost.for.two, na.rm = TRUE)) %>%
  arrange(desc(Avg_Cost)) %>%
  head(5)
print(expensive_cities)
```

```
> expensive_cities <- data %>%
+   group_by(City) %>%
+   summarise(Avg_Cost = mean(Average.Cost.for.two, na.rm = TRUE)) %>%
+   arrange(desc(Avg_Cost)) %>%
+   head(5)
> print(expensive_cities)
# A tibble: 5 × 2
  City      Avg_Cost
  <chr>        <dbl>
1 Panchkula     2000
2 Hyderabad     1361.
3 Pune          1338.
4 Jaipur        1310
5 Kolkata       1272.
```

LEVEL - 5

```
# 13. Create a new column PriceCategory
# Create the 'PriceCategory' column
data <- data %>%
  mutate(PriceCategory = case_when(
    Average.Cost.for.two < 300 ~ "Low",
    Average.Cost.for.two >= 300 & Average.Cost.for.two <= 800 ~
"Medium",
    Average.Cost.for.two > 800 ~ "High"
  ))
```

```
# Print first 10 rows with Restaurant Name, City, Cost, and Price Category
head(select(data, Restaurant.Name, City, Average.Cost.for.two,
PriceCategory),10)
```

```
> data <- data %>%
+   mutate(PriceCategory = case_when(
+     Average.Cost.for.two < 300 ~ "Low",
+     Average.Cost.for.two >= 300 & Average.Cost.for.two <= 800 ~ "Medium",
+     Average.Cost.for.two > 800 ~ "High"
+   ))
> head(select(data, Restaurant.Name, City, Average.Cost.for.two, PriceCategory), 10)
Error: unexpected invalid token in "head(select(data, Restaurant.Name, City, Average.Cost.for.two, PriceCategory), "
> head(select(data, Restaurant.Name, City, Average.Cost.for.two, PriceCategory),10)
# A tibble: 10 × 4
   Restaurant.Name                     City  Average.Cost.for.two PriceCategory
   <chr>                               <chr>               <dbl> <chr>
 1 Jahanpanah                          Agra                  850 High
 2 Rangrezz Restaurant                 Agra                  700 Medium
 3 Time2Eat - Mama Chicken             Agra                  500 Medium
 4 Chokho Jeeman Marwari Jain Bhojanalya Agra                400 Medium
 5 Pinch Of Spice                      Agra                 1000 High
 6 MoMo Cafe                           Agra                 2000 High
 7 Peshawri - ITC Mughal               Agra                 2500 High
 8 Taj Bano - ITC Mughal               Agra                 2500 High
 9 G Thal                              Agra                  800 Medium
10 Dawat-e-Nawab - Radisson Blu        Agra                 3600 High
```

```
# 14. Create PopularityTag based on Rating and Votes
data <- data %>%
  mutate(PopularityTag = ifelse(Aggregate.rating > 4 & Votes > 500,
"Popular", "Regular"))

# Compute the percentage of popular restaurants in each city
popular_percentage <- data %>%
  group_by(City) %>%
  summarise(
    Total = n(),
    Popular = sum(PopularityTag == "Popular"),
    Percentage_Popular = round((Popular / Total) * 100, 2)
  ) %>%
  arrange(desc(Percentage_Popular))

print(head(popular_percentage, 10))

# Show top 10 cities by popularity percentage
```

```
> data <- data %>%
+   mutate(PopularityTag = ifelse(Aggregate.rating > 4 & Votes > 500, "Popular", "Regular"))
>
> # Compute the percentage of popular restaurants in each city
> popular_percentage <- data %>%
+   group_by(City) %>%
+   summarise(
+     Total = n(),
+     Popular = sum(PopularityTag == "Popular"),
+     Percentage_Popular = round((Popular / Total) * 100, 2)
+   ) %>%
+   arrange(desc(Percentage_Popular))
> print(head(popular_percentage, 10))  # Show top 10 cities by popularity percentage
# A tibble: 10 × 4
   City         Total Popular Percentage_Popular
   <chr>        <int>   <int>              <dbl>
 1 Panchkula        1       1              100
 2 Chennai         20      15               75
 3 Bangalore       20      14               70
 4 Kolkata         20      14               70
 5 Hyderabad       18      11               61.1
 6 Pune            20      11               55
 7 Secunderabad     2       1               50
 8 Goa             20       8               40
 9 Mumbai          20       8               40
10 Ahmedabad       21       8               38.1
```

WHOLE DATASET INTERPRETETION

Metro Dominance: Major cities like New Delhi, Bangalore, and Mumbai dominate the restaurant count, with New Delhi having the highest number of listings.

Cuisine & Affordability: North Indian, Chinese, and Fast Food are the most common cuisines. A significant number of restaurants offer good food at affordable prices (under ₹300) with ratings above 4.0.

Customer Preferences: Only a small portion of restaurants offer both online delivery and table booking, showing that most focus on one service type based on demand.

Quality & Popularity: Popular restaurants (high ratings and votes) are mostly located in metros. The average rating is 2.52, suggesting many restaurants still have room to improve quality.

Segmented Pricing: Most restaurants fall in the Medium (₹300–₹800) price category. Feature engineering reveals New Delhi also leads in the percentage of "Popular" restaurants, highlighting it as a hotspot for both quantity and quality.

Conclusion

The **Zomato India restaurant dataset** offers deep **insights** into the country's dining landscape. Metro cities, especially **New Delhi**, emerge as culinary hubs with both high **quantity** and **quality** of restaurants.
While **North Indian** and **Chinese cuisines** dominate the menu, there's a promising presence of **budget-friendly** yet **highly-rated restaurants**, proving that quality dining doesn't always come at a premium. The majority of restaurants fall into the **medium price range**, catering to average consumer spending habits. However, only a fraction provide both online delivery and table booking, indicating scope for service diversification. With data-driven categorization like **PriceCategory** and **PopularityTag**, this analysis equips businesses and foodies alike with a smarter approach to navigating India's vibrant food scene.

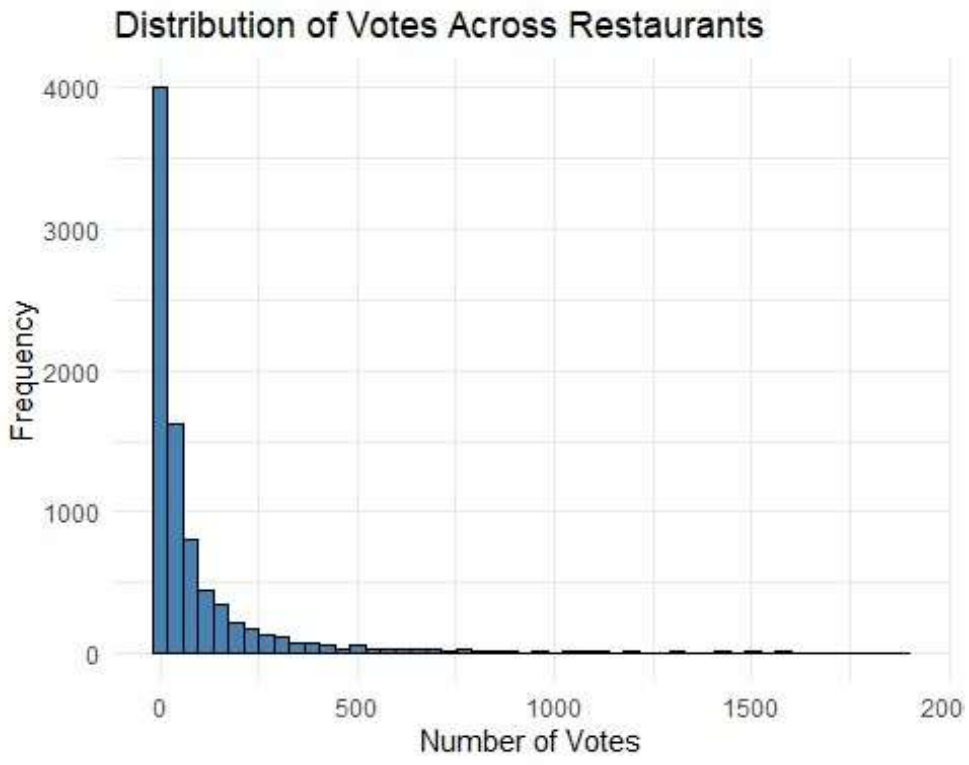
Project Title: “ Data Analysis of Zomato India Restaurant Dataset”

6. Visualizations

Project Link - https://rpubs.com/Adityasri8626/1302988

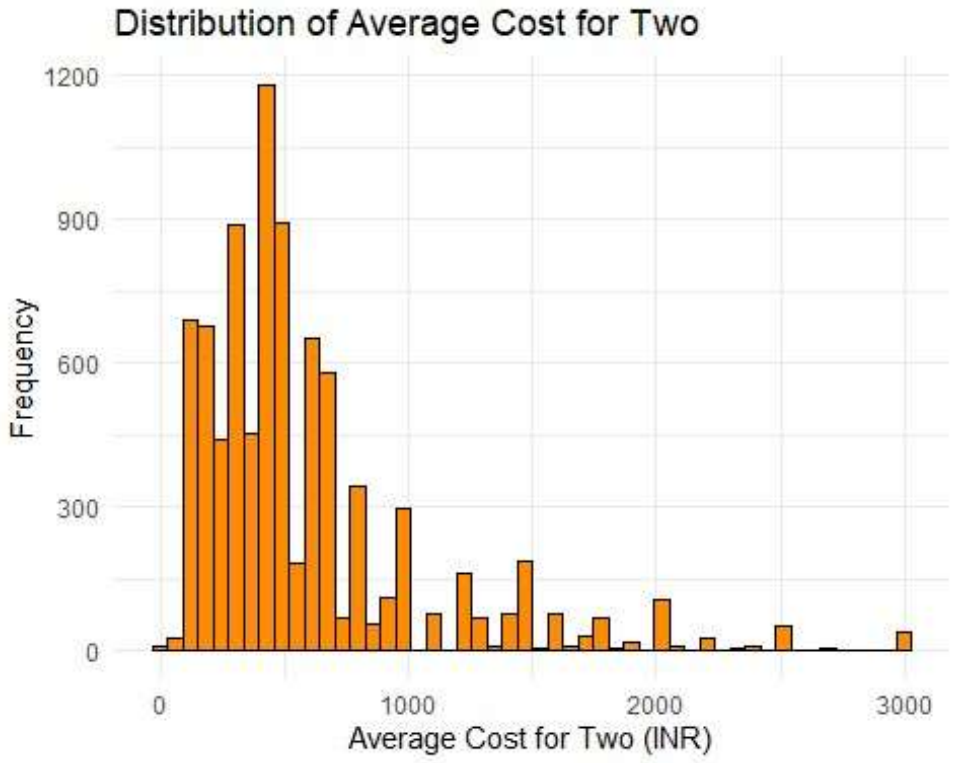
6.1 Distribution of Votes

```
filtered_data <- data %>% filter(Votes <= quantile(Votes, 0.99, na.rm =
TRUE))

ggplot(filtered_data, aes(x = Votes)) +
  geom_histogram(bins = 50, fill = "steelblue", color = "black") +
  labs(title = "Distribution of Votes Across Restaurants", x = "Number of
Votes", y = "Frequency") +
  theme_minimal()
```



6.2 Distribution of Cost

```
filtered_cost_data <- data %>% filter(Average.Cost.for.two <=
quantile(Average.Cost.for.two, 0.99, na.rm = TRUE))

ggplot(filtered_cost_data, aes(x = Average.Cost.for.two)) +
  geom_histogram(bins = 50, fill = "darkorange", color = "black") +
  labs(title = "Distribution of Average Cost for Two", x = "Average Cost for
Two (INR)", y = "Frequency") +
  theme_minimal()
```



7. Modeling

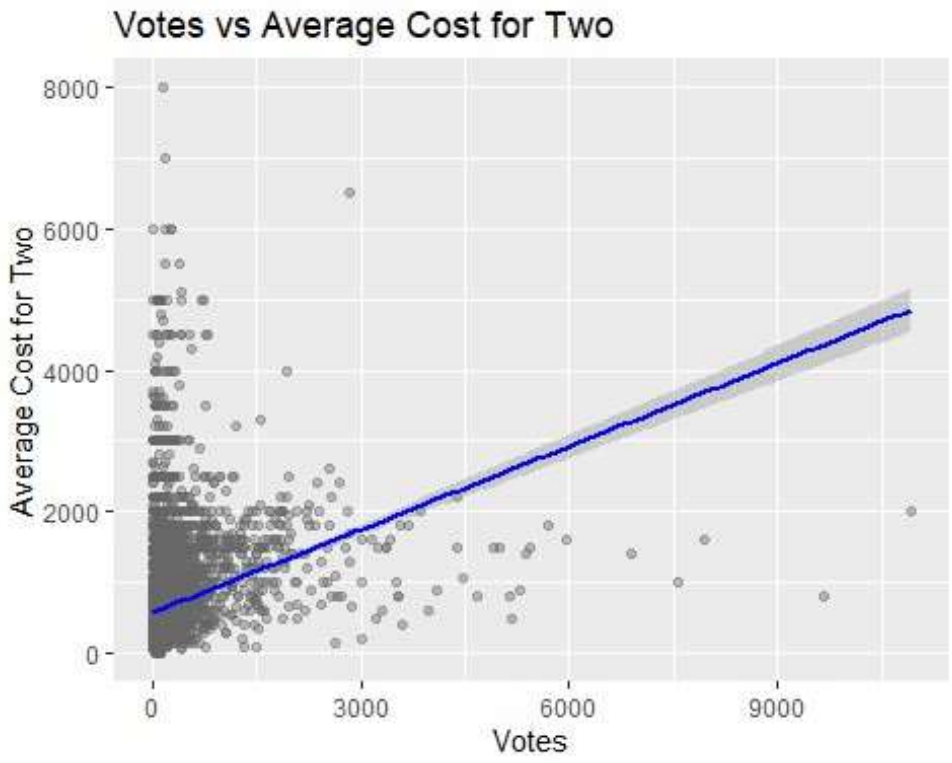
7.1 Simple Linear Regression: Votes vs Cost

```
simple_model <- lm(Average.Cost.for.two ~ Votes, data = data)
summary(simple_model)

##
## Call:
## lm(formula = Average.Cost.for.two ~ Votes, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3560.1  -283.7  -139.0    78.9  7373.6
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 569.56724    6.45293   88.27   <2e-16 ***
## Votes         0.39211    0.01436   27.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 571.6 on 8650 degrees of freedom
## Multiple R-squared:  0.07939,    Adjusted R-squared:  0.07929
## F-statistic:   746 on 1 and 8650 DF,  p-value: < 2.2e-16
```

```r
# Plot
ggplot(data, aes(x = Votes, y = Average.Cost.for.two)) +
  geom_point(alpha = 0.4, color = "gray40") +
  geom_smooth(method = "lm", se = TRUE, color = "blue") +
  labs(title = "Votes vs Average Cost for Two", x = "Votes", y = "Average
Cost for Two")
```

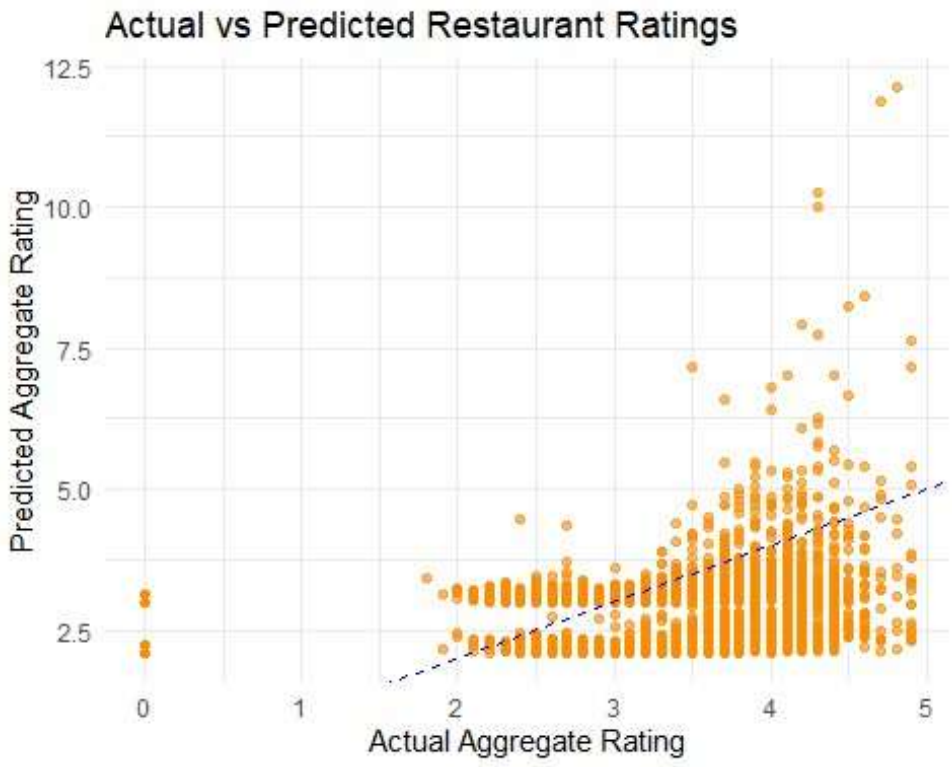
```
## `geom_smooth()` using formula = 'y ~ x'
```



7.2 Multiple Linear Regression: Rating

```r
data$OnlineDelivery <- ifelse(data$Has.Online.delivery == "Yes", 1, 0)

tier1_cities <- c("Delhi", "New Delhi", "Mumbai", "Bangalore", "Bengaluru")
tier2_cities <- c("Hyderabad", "Chennai", "Kolkata", "Pune")
```

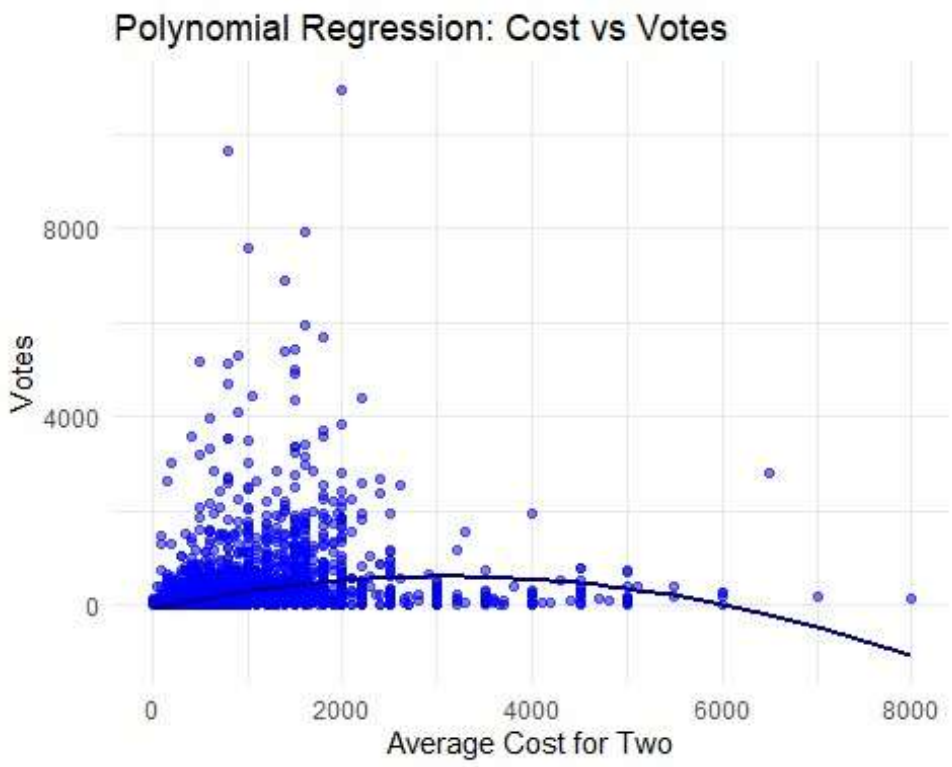
```r
data$CityTier <- case_when(
  data$City %in% tier1_cities ~ 1,
  data$City %in% tier2_cities ~ 2,
  TRUE ~ 3
)

multi_model_final <- lm(Aggregate.rating ~ Votes + OnlineDelivery + CityTier,
data = data)
summary(multi_model_final)

##
## Call:
## lm(formula = Aggregate.rating ~ Votes + OnlineDelivery + CityTier,
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3257 -1.6020  0.5276  1.0414  2.6337
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.013e+00  3.221e-02  62.493  < 2e-16 ***
## Votes          9.178e-04  3.506e-05  26.178  < 2e-16 ***
## OnlineDelivery 8.992e-01  3.343e-02  26.900  < 2e-16 ***
## CityTier       7.733e-02  1.564e-02   4.944 7.81e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.388 on 8648 degrees of freedom
## Multiple R-squared:  0.1563, Adjusted R-squared:  0.156
## F-statistic: 533.9 on 3 and 8648 DF,  p-value: < 2.2e-16

data$Predicted_Rating_Final <- predict(multi_model_final, newdata = data)

ggplot(data, aes(x = Aggregate.rating, y = Predicted_Rating_Final)) +
  geom_point(color = "darkorange", alpha = 0.6) +
  geom_abline(intercept = 0, slope = 1, color = "blue", linetype = "dashed")
+
  labs(title = "Actual vs Predicted Restaurant Ratings", x = "Actual
Aggregate Rating", y = "Predicted Aggregate Rating") +
  theme_minimal()
```



7.3 Polynomial Regression: Votes vs Cost

```
poly_cost_votes_model <- lm(Votes ~ poly(Average.Cost.for.two, 2, raw =
TRUE), data = data)
summary(poly_cost_votes_model)

##
## Call:
## lm(formula = Votes ~ poly(Average.Cost.for.two, 2, raw = TRUE),
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -602.9  -112.6   -40.9    14.6 10420.0
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                -8.472e+01  8.286e+00  -10.22
## poly(Average.Cost.for.two, 2, raw = TRUE)1  4.397e-01  1.525e-02   28.83
## poly(Average.Cost.for.two, 2, raw = TRUE)2 -7.017e-05  3.964e-06  -17.70
##                                            Pr(>|t|)
## (Intercept)                                  <2e-16 ***
## poly(Average.Cost.for.two, 2, raw = TRUE)1   <2e-16 ***
## poly(Average.Cost.for.two, 2, raw = TRUE)2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 403.5 on 8649 degrees of freedom
## Multiple R-squared:  0.1116, Adjusted R-squared:  0.1114
## F-statistic: 543.2 on 2 and 8649 DF,  p-value: < 2.2e-16

data$Predicted_Votes_Cost <- predict(poly_cost_votes_model, newdata = data)

ggplot(data, aes(x = Average.Cost.for.two, y = Votes)) +
  geom_point(color = "blue", alpha = 0.5) +
  geom_line(aes(y = Predicted_Votes_Cost), color = "darkblue", linewidth = 1)
+
  labs(title = "Polynomial Regression: Cost vs Votes", x = "Average Cost for
Two", y = "Votes") +
  theme_minimal()
```



8. Statistical Tests

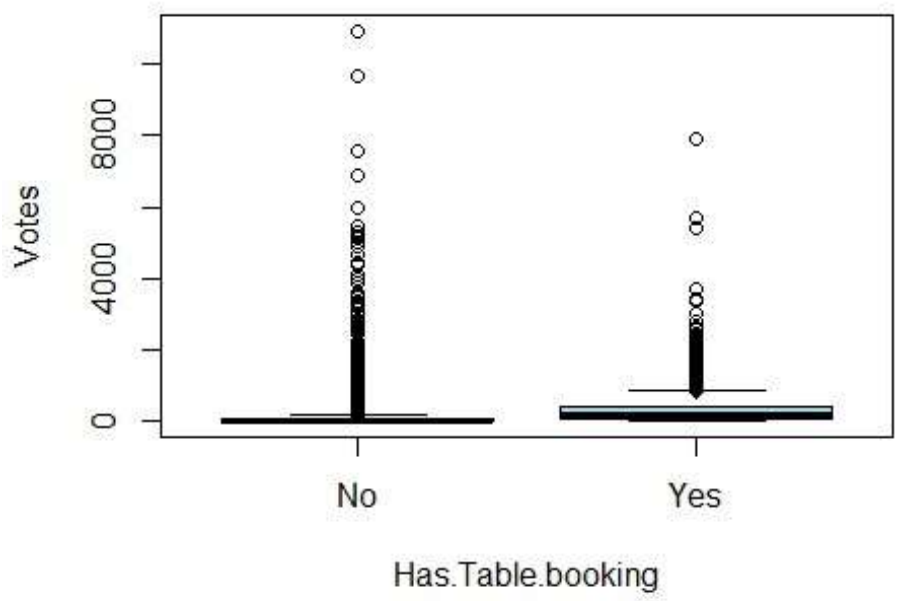
8.1 ANOVA: Online Delivery vs Rating

```
data$Has.Online.delivery <- as.factor(data$Has.Online.delivery)
anova_result <- aov(Aggregate.rating ~ Has.Online.delivery, data = data)
summary(anova_result)

##                      Df Sum Sq Mean Sq F value Pr(>F)
## Has.Online.delivery    1   1723  1723.0   826.7 <2e-16 ***
## Residuals           8650  18028     2.1
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

boxplot(Aggregate.rating ~ Has.Online.delivery, data = data, col =
c("tomato", "skyblue"))
```

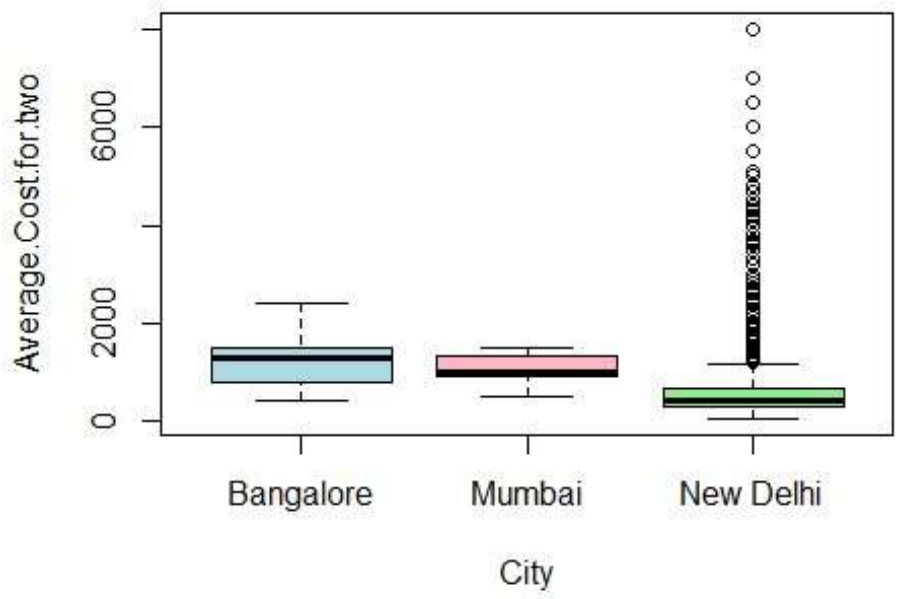
8.2 ANOVA: Table Booking vs Votes

```
data$Has.Table.booking <- as.factor(data$Has.Table.booking)
anova_votes <- aov(Votes ~ Has.Table.booking, data = data)
summary(anova_votes)

##                     Df    Sum Sq  Mean Sq F value Pr(>F)
## Has.Table.booking    1 5.606e+07 56056876   317.1 <2e-16 ***
## Residuals         8650 1.529e+09   176758
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

boxplot(Votes ~ Has.Table.booking, data = data, col = c("lightcoral",
"lightblue"))
```



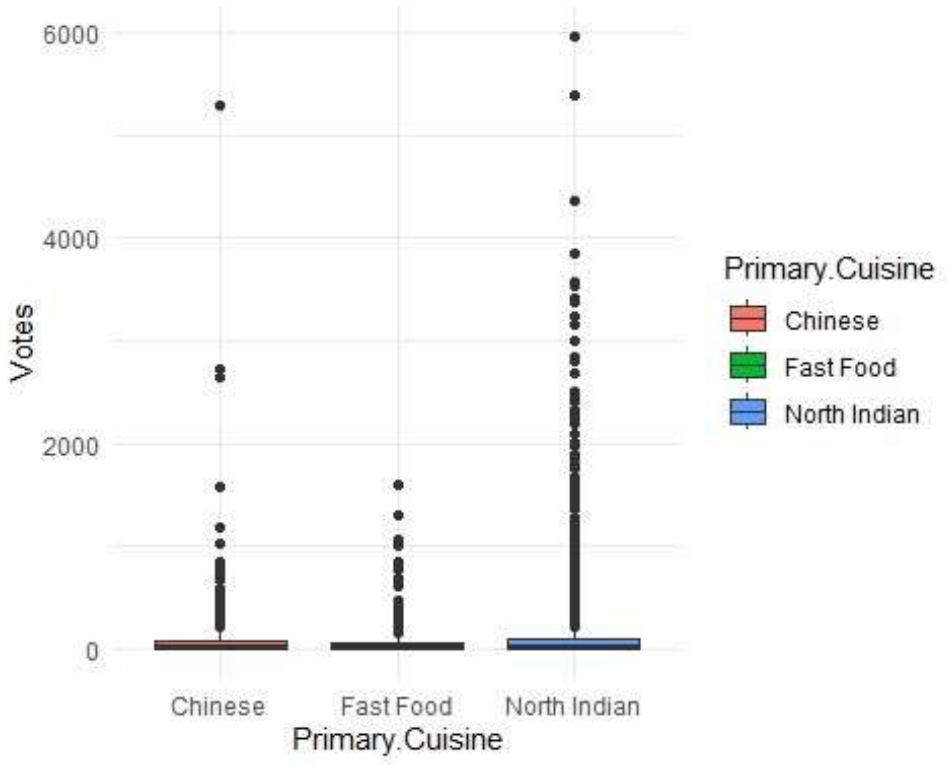
8.3 ANOVA: Average Cost across Cities

```
selected_data <- subset(data, City %in% c("Mumbai", "New Delhi",
"Bangalore"))
selected_data <- selected_data[!is.na(selected_data$Average.Cost.for.two), ]
selected_data$City <- factor(selected_data$City)

anova_city <- aov(Average.Cost.for.two ~ City, data = selected_data)
summary(anova_city)

##               Df    Sum Sq Mean Sq F value   Pr(>F)
## City           2 1.255e+07 6274959   15.61 1.74e-07 ***
## Residuals   5510 2.215e+09  402030
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

boxplot(Average.Cost.for.two ~ City, data = selected_data, col =
c("lightblue", "lightpink", "lightgreen"))
```



8.4 ANOVA: Cuisine Type vs Votes

```
data$Primary.Cuisine <- sapply(strsplit(as.character(data$Cuisines), ","),
`[`, 1)
data$Primary.Cuisine <- trimws(data$Primary.Cuisine)

top3_cuisines <- data %>% count(Primary.Cuisine, sort = TRUE) %>% top_n(3, n)
%>% pull(Primary.Cuisine)
filtered_data <- data %>% filter(Primary.Cuisine %in% top3_cuisines)

anova_top3 <- aov(Votes ~ Primary.Cuisine, data = filtered_data)
summary(anova_top3)

##                   Df    Sum Sq Mean Sq F value   Pr(>F)
## Primary.Cuisine    2   1992269  996134   9.859 5.34e-05 ***
## Residuals       4474 452042276  101038
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

ggplot(filtered_data, aes(x = Primary.Cuisine, y = Votes, fill =
Primary.Cuisine)) +
  geom_boxplot() +
  theme_minimal()
```



9. Correlation Analysis

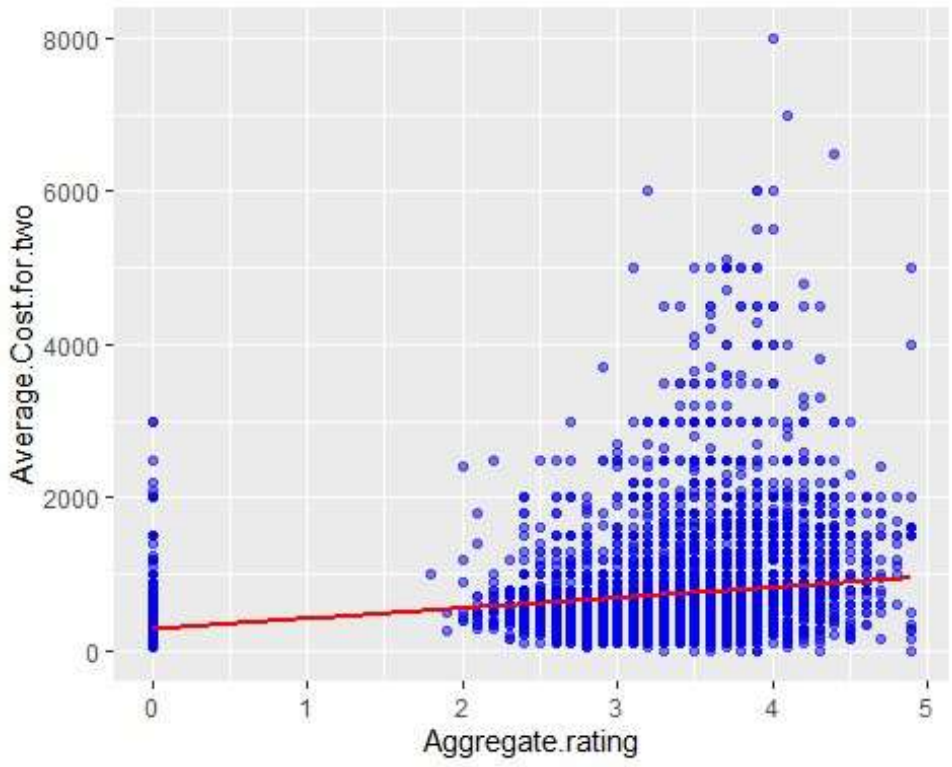
9.1 Rating vs Cost

```
cor(data$Aggregate.rating, data$Average.Cost.for.two, use = "complete.obs")
```

```
## [1] 0.3441716
```

```
ggplot(data, aes(x = Aggregate.rating, y = Average.Cost.for.two)) +
  geom_point(color = "blue", alpha = 0.5) +
  geom_smooth(method = "lm", color = "red", se = FALSE)
```

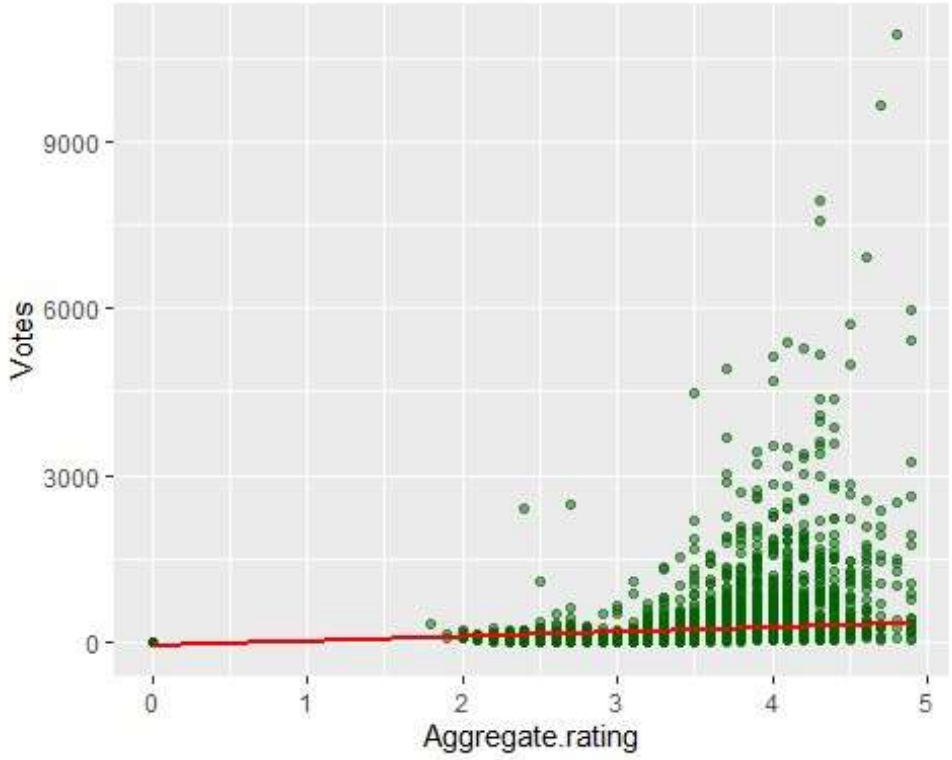
```
## `geom_smooth()` using formula = 'y ~ x'
```



9.2 Rating vs Votes

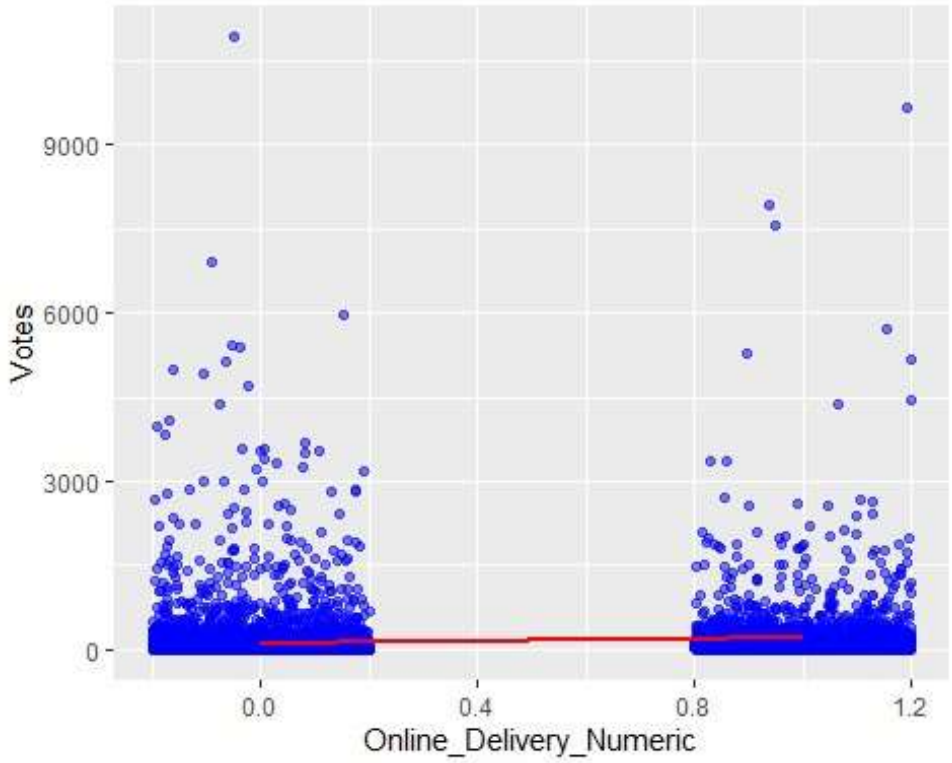
```
correlation_value <- cor(data$Aggregate.rating, data$Votes, use =
"complete.obs")
cor_test_result <- cor.test(data$Aggregate.rating, data$Votes)

print(correlation_value)

## [1] 0.2876924

print(cor_test_result)

##
##  Pearson's product-moment correlation
##
## data:  data$Aggregate.rating and data$Votes
## t = 27.938, df = 8650, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2682468 0.3069037
## sample estimates:
##       cor
## 0.2876924

ggplot(data, aes(x = Aggregate.rating, y = Votes)) +
  geom_point(color = "darkgreen", alpha = 0.5) +
  geom_smooth(method = "lm", color = "red", se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



9.3 Online Delivery vs Votes

```
data$Online_Delivery_Numeric <- ifelse(data$Has.Online.delivery == "Yes", 1,
0)
correlation_value <- cor(data$Online_Delivery_Numeric, data$Votes, use =
"complete.obs")

print(correlation_value)

## [1] 0.1047307

ggplot(data, aes(x = Online_Delivery_Numeric, y = Votes)) +
  geom_jitter(width = 0.2, height = 0, alpha = 0.5, color = "blue") +
  geom_smooth(method = "lm", color = "red", se = FALSE)

## `geom_smooth()` using formula = 'y ~ x'
```


9000
6000
3000
0
Votes
0.0
0.4
0.8
1.2
Online_Delivery_Numeric

WHOLE DATASET INTERPRETETION

- The dataset contained **8652 records** and **21 features**, with no missing values.
- The **average restaurant rating** was **2.52** and the **average cost for two** was around **₹623**.
- **New Delhi**, **Gurgaon**, and **Noida** had the highest number of listed restaurants.
- **High-rated and low-cost restaurants** were mainly located in **Agra** and **Amritsar**.
- A few restaurants offered **both table booking and online delivery**, primarily located in **Bangalore** and **Chennai**.
- **North Indian**, **Chinese**, and **Fast Food** were the most common cuisines.
- **New Delhi**, **Gurgaon**, and **Noida** led in cities with the most **highly rated** restaurants.
- **Barbeque Nation** (Kolkata) and **Absolute Barbecues** (Hyderabad) were among the **top popular restaurants**.
- **Affordable high-rated options** were available, especially in cities like **Agra** and **Amritsar**.
- **Panchkula** was the **most expensive city** based on average restaurant cost.
- Feature engineering classified restaurants into **Low**, **Medium**, and **High** price categories and tagged **Popular** ones.
- Visualizations showed that **votes** and **cost distributions**
- **Simple and multiple linear regression** models showed significant but moderate relationships between votes, cost, and ratings.
- **ANOVA tests** indicated significant differences in ratings based on **online delivery** availability, **table booking**, and between **cities**.
- Correlation analysis found **moderate positive correlations** between rating and votes (0.29) and between rating and cost (0.34).