Kaggle project DSCI 478

2025-02-19

LOAD DATA SET

```r
# Read Titanic data set
train_df <- read.csv("train.csv", stringsAsFactors = FALSE)
test_df <- read.csv("test.csv", stringsAsFactors = FALSE)

head(train_df)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                  Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                            Allen, Mr. William Henry   male  35     0     0
## 6                                    Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500              S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250              S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500              S
## 6           330877  8.4583              Q
```

```r
head(test_df)
```

```
##   PassengerId Pclass                                         Name    Sex  Age
## 1         892      3                             Kelly, Mr. James   male 34.5
## 2         893      3             Wilkes, Mrs. James (Ellen Needs) female 47.0
## 3         894      2                    Myles, Mr. Thomas Francis   male 62.0
## 4         895      3                             Wirz, Mr. Albert   male 27.0
## 5         896      3 Hirvonen, Mrs. Alexander (Helga E Lindqvist) female 22.0
## 6         897      3                   Svensson, Mr. Johan Cervin   male 14.0
##   SibSp Parch  Ticket    Fare Cabin Embarked
## 1     0     0  330911  7.8292              Q
## 2     1     0  363272  7.0000              S
## 3     0     0  240276  9.6875              Q
## 4     0     0  315154  8.6625              S
## 5     1     1 3101298 12.2875              S
## 6     0     0    7538  9.2250              S
```

CHECK FOR MISSING VALUES

```r
# Count missing values in each column
colSums(is.na(train_df))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

```r
# Remove rows where Age is missing
train_df <- train_df[!is.na(train_df$Age), ]
```

DOUBLE CHECK TO MAKE SURE THERE ARE NO MORE MISSING VALUES AFTER REMOVING ROWS TO BE DONE CORRECTLY

```r
# Count missing values in each column
colSums(is.na(train_df))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

CONVERT CATEGORICAL VARIABLES TO NUMERIC

```r
# Sex: Male -> 0; Females -> 1
train_df$Sex <- ifelse(train_df$Sex == "male", 0, 1)
test_df$Sex <- ifelse(test_df$Sex == "male", 0, 1)

# Embarked: "C" (Cherbourg) -> 0; "Q" (Queenstown) -> 1; "S" (Southampton) -> 2
train_df$Embarked <- ifelse(train_df$Embarked == "C", 0,
                            ifelse(train_df$Embarked == "Q", 1, 2))
test_df$Embarked <- ifelse(test_df$Embarked == "C", 0,
                           ifelse(test_df$Embarked == "Q", 1, 2))

# Fare: Round to two decimal places
train_df$Fare <- round(train_df$Fare, 2)
test_df$Fare <- round(test_df$Fare, 2)

# Check the first few rows to confirm changes
head(train_df[, c("PassengerId", "Survived", "Pclass", "Sex", "Age", "SibSp", "Parch", "Fare", "Embarke
```

```
##   PassengerId Survived Pclass Sex Age SibSp Parch  Fare Embarked
## 1           1        0      3   0  22     1     0  7.25        2
## 2           2        1      1   1  38     1     0 71.28        0
## 3           3        1      3   1  26     0     0  7.92        2
## 4           4        1      1   1  35     1     0 53.10        2
## 5           5        0      3   0  35     0     0  8.05        2
## 7           7        0      1   0  54     0     0 51.86        2
```

SAVE CLEAN DATASET

```r
write.csv(train_df, "clean_train.csv", row.names = FALSE)
write.csv(test_df, "clean_test.csv", row.names = FALSE)

head(train_df)
```

```
##   PassengerId Survived Pclass
```

```
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 7           7        0      1
##                                                   Name Sex Age SibSp Parch
## 1                              Braund, Mr. Owen Harris   0  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   1  38     1     0
## 3                               Heikkinen, Miss. Laina   1  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel)   1  35     1     0
## 5                             Allen, Mr. William Henry   0  35     0     0
## 7                              McCarthy, Mr. Timothy J   0  54     0     0
##             Ticket  Fare Cabin Embarked
## 1        A/5 21171  7.25              2
## 2         PC 17599 71.28   C85        0
## 3 STON/O2. 3101282  7.92              2
## 4           113803 53.10  C123        2
## 5           373450  8.05              2
## 7            17463 51.86   E46        2
```

LOADING CLEAN DATASET

```r
# Load the cleaned dataset
train_df <- read.csv("clean_train.csv", stringsAsFactors = FALSE)
test_df <- read.csv("clean_test.csv", , stringsAsFactors = FALSE)

# Check the first few rows to verify correctness
head(train_df)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           7        0      1
##                                                   Name Sex Age SibSp Parch
## 1                              Braund, Mr. Owen Harris   0  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   1  38     1     0
## 3                               Heikkinen, Miss. Laina   1  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel)   1  35     1     0
## 5                             Allen, Mr. William Henry   0  35     0     0
## 6                              McCarthy, Mr. Timothy J   0  54     0     0
##             Ticket  Fare Cabin Embarked
## 1        A/5 21171  7.25              2
## 2         PC 17599 71.28   C85        0
## 3 STON/O2. 3101282  7.92              2
## 4           113803 53.10  C123        2
## 5           373450  8.05              2
## 6            17463 51.86   E46        2
```

```r
colSums(is.na(train_df))
```

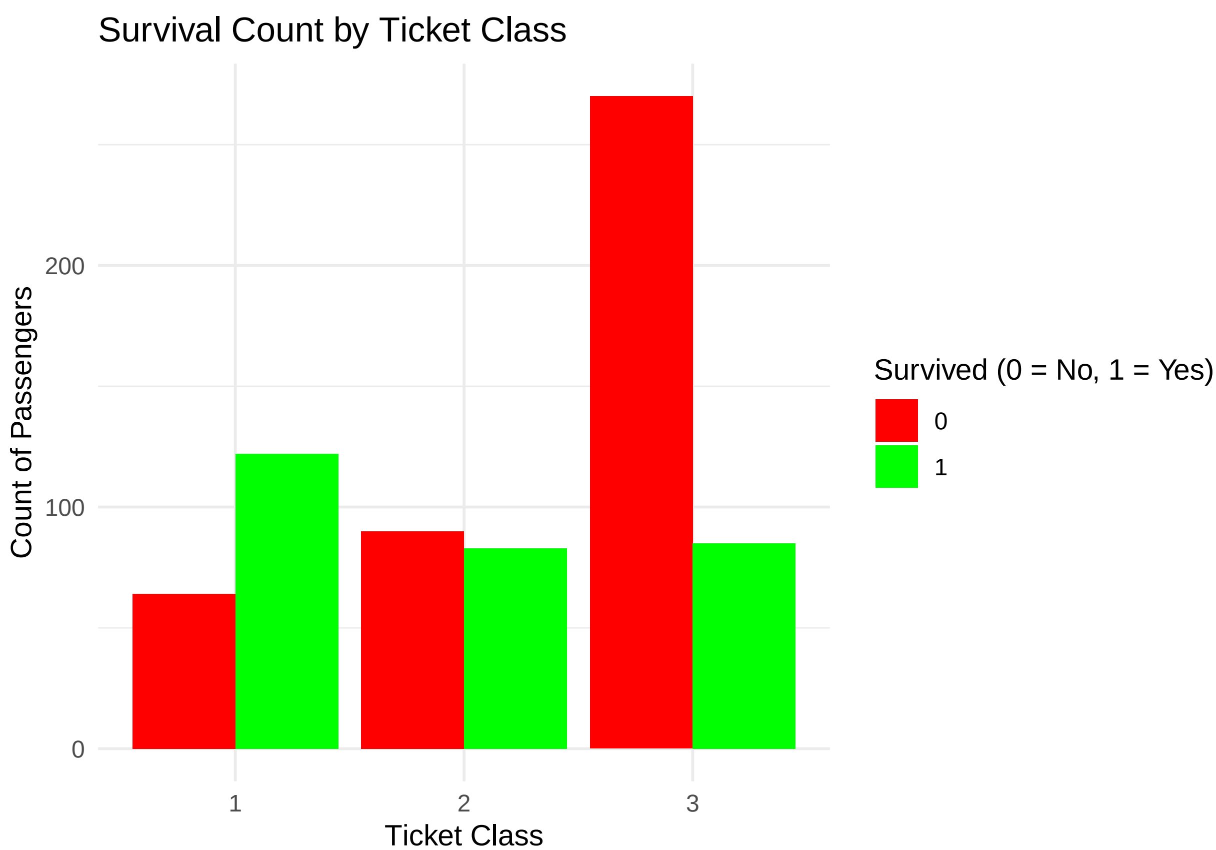
```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
```

```
##          SibSp        Parch       Ticket         Fare        Cabin     Embarked
##              0            0            0            0            0            0
```

EDA

Ticket Class (Pclass)

```r
#Analyzing survival rates across different ticket classes.
# Survival Rate by Ticket Class
ggplot(train_df, aes(x = as.factor(Pclass), fill = as.factor(Survived))) +
  geom_bar(position = "dodge") +
  labs(title = "Survival Count by Ticket Class",
       x = "Ticket Class",
       y = "Count of Passengers") +
  scale_fill_manual(values = c("red", "green"), name = "Survived (0 = No, 1 = Yes)") +
  theme_minimal()
```



Insights from the Plot:

1. First Class (Ticket Class 1):
   - More survivors than non-survivors.

   - Indicates a high survival rate among upper-class passengers.
2. Second Class (Ticket Class 2):

    - The number of survivors and non-survivors is relatively balanced.

    - Suggests that survival was roughly equal among second-class passengers.
3. Third Class (Ticket Class 3):
    - Significantly more non-survivors than survivors.

    - Suggests that lower-class passengers had a much lower chance of survival.

Inference:

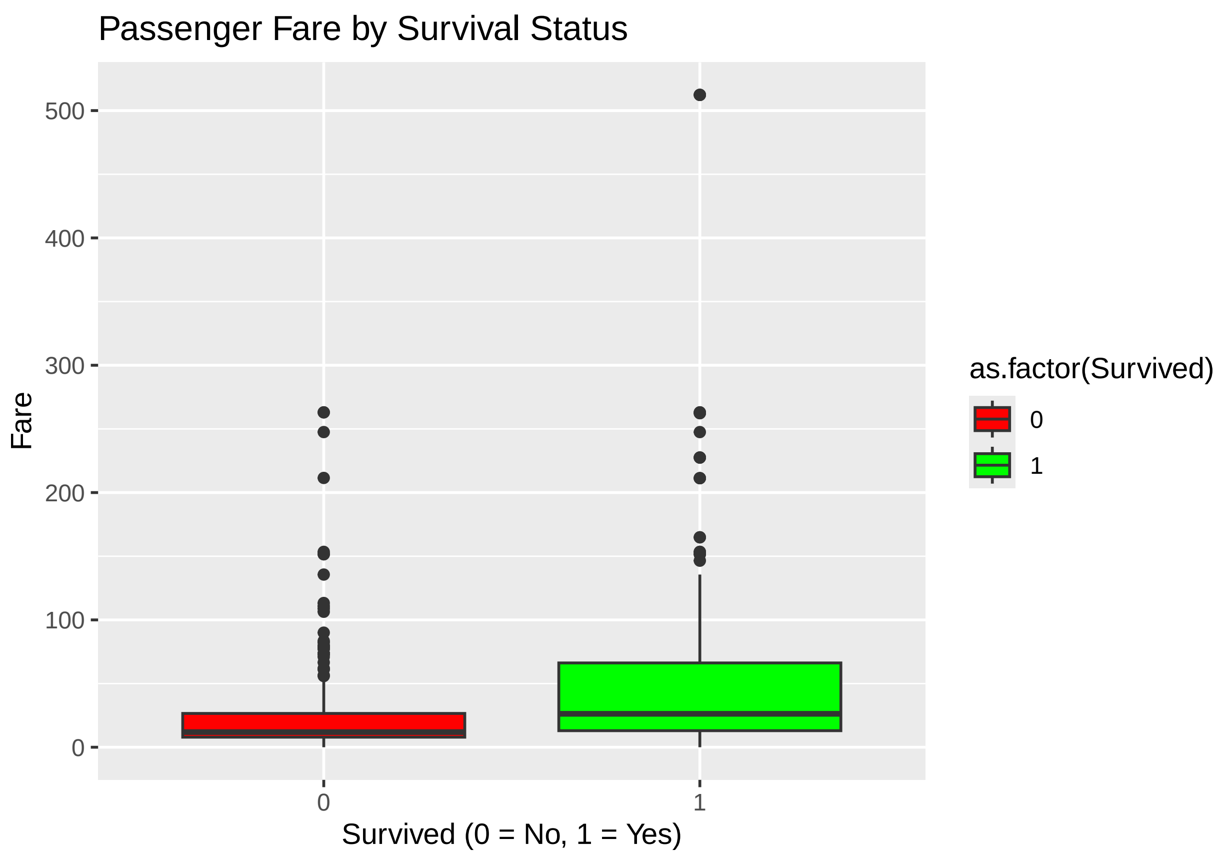
- There was a clear class disparity in survival rates on the Titanic.

- First-class passengers had the highest survival rates, likely due to better access to lifeboats and being prioritized during evacuation.

- Third-class passengers faced the highest mortality rates, potentially due to being located in lower decks and having limited access to lifeboats.

Passenger Fair (Fair)

```r
# Fare distribution by survival status
ggplot(train_df, aes(x = as.factor(Survived), y = Fare, fill = as.factor(Survived))) +
  geom_boxplot() +
  labs(title = "Passenger Fare by Survival Status", x = "Survived (0 = No, 1 = Yes)", y = "Fare") +
  scale_fill_manual(values = c("red", "green"))
```



Insights from the Plot:

1. Passengers Who Did Not Survive (0):
   - Most non-survivors paid lower fares.
   - The majority of fares cluster around lower amounts, with a few outliers who paid higher fares but still did not survive.
2. Passengers Who Survived (1):
   - Survivors generally paid higher fares compared to non-survivors.
   - There are several high-value outliers where passengers who paid extremely high fares also survived.

Inference:

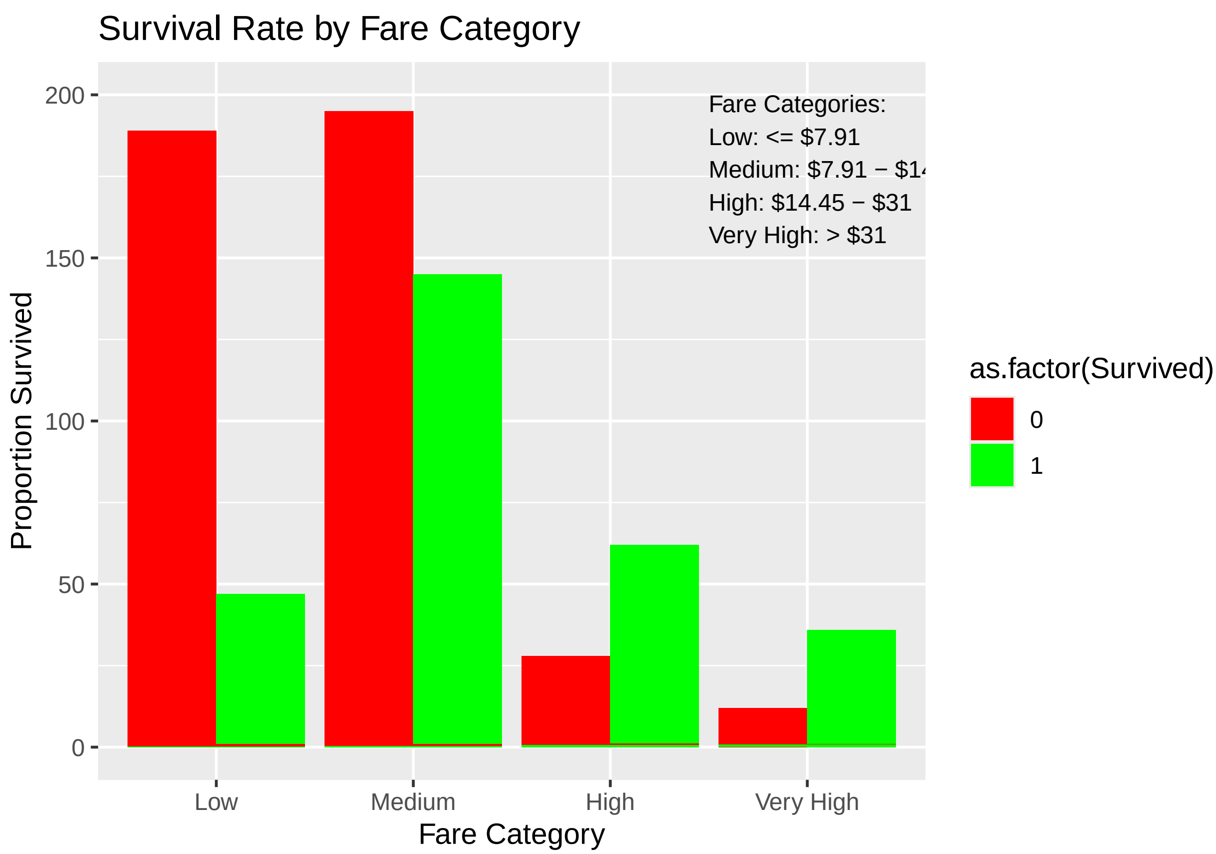
- There appears to be a positive relationship between fare and survival: passengers who paid higher fares were more likely to survive.
- Higher fares likely correlate with higher social class (first-class tickets), offering better access to lifeboats and safer areas on the ship.
- Some passengers who paid high fares did not survive, indicating that fare alone was not the sole factor influencing survival.

Categorizing Fares

```r
## fare categories to compare

train_df$Fare_Category <- cut(train_df$Fare,
                              breaks = c(-1, 10, 50, 100, 600),
                              labels = c("Low", "Medium", "High", "Very High"))

# survival rate by fare category
ggplot(train_df, aes(x = Fare_Category, fill = as.factor(Survived))) +
  geom_bar(position = "dodge") +
  geom_bar(position = "fill") +
  labs(title = "Survival Rate by Fare Category", x = "Fare Category", y = "Proportion Survived") +
  annotate("text", x = 3.5, y = 200, label = "Fare Categories:\nLow: <= $7.91\nMedium: $7.91 - $14.45\n
           hjust = 0, vjust = 1, size = 3, color = "black") +
  scale_fill_manual(values = c("red", "green"))
```



Insights from the Plot:

1. Low Fare Category:
   - A significantly higher number of passengers did not survive.

   - Indicates that passengers who paid the lowest fares had the lowest survival rates.
2. Medium Fare Category:
   - A noticeable increase in survival rate compared to the low fare group.

- Many passengers in this category survived, suggesting moderate fares had a positive impact on survival chances.

3. High Fare Category:
   - More passengers survived than those who did not.
   - Shows a clear trend where higher fares were associated with a greater chance of survival.
4. Very High Fare Category:
   - A small number of passengers, but most survived.
   - Suggests that passengers who paid very high fares had the highest likelihood of survival, likely due to better access to lifeboats and priority treatment.

Inference:

- There is a clear positive correlation between fare category and survival rate: the higher the fare paid, the greater the likelihood of survival.
- This supports the idea that wealth and social class played a significant role in determining who survived the Titanic disaster.
- Passengers paying very high fares had an advantage, likely due to better accommodations and quicker access to safety measures during the evacuation.

Feature Engineering

FarePerClass Feature

```r
# FarePerClass feature
train_df$FarePerClass <- train_df$Fare / train_df$Pclass
test_df$FarePerClass <- test_df$Fare / test_df$Pclass

# new feature
head(train_df[, c("Pclass", "Fare", "FarePerClass")])
```

```
##   Pclass  Fare FarePerClass
## 1      3  7.25     2.416667
## 2      1 71.28    71.280000
## 3      3  7.92     2.640000
## 4      1 53.10    53.100000
## 5      3  8.05     2.683333
## 6      1 51.86    51.860000
```

The table is the FarePerClass feature, which is calculated by dividing the fare each passenger paid by their ticket class (Pclass). This feature helps normalize fare values based on the passenger class, allowing the model to better understand the relative expense of a ticket within each class.

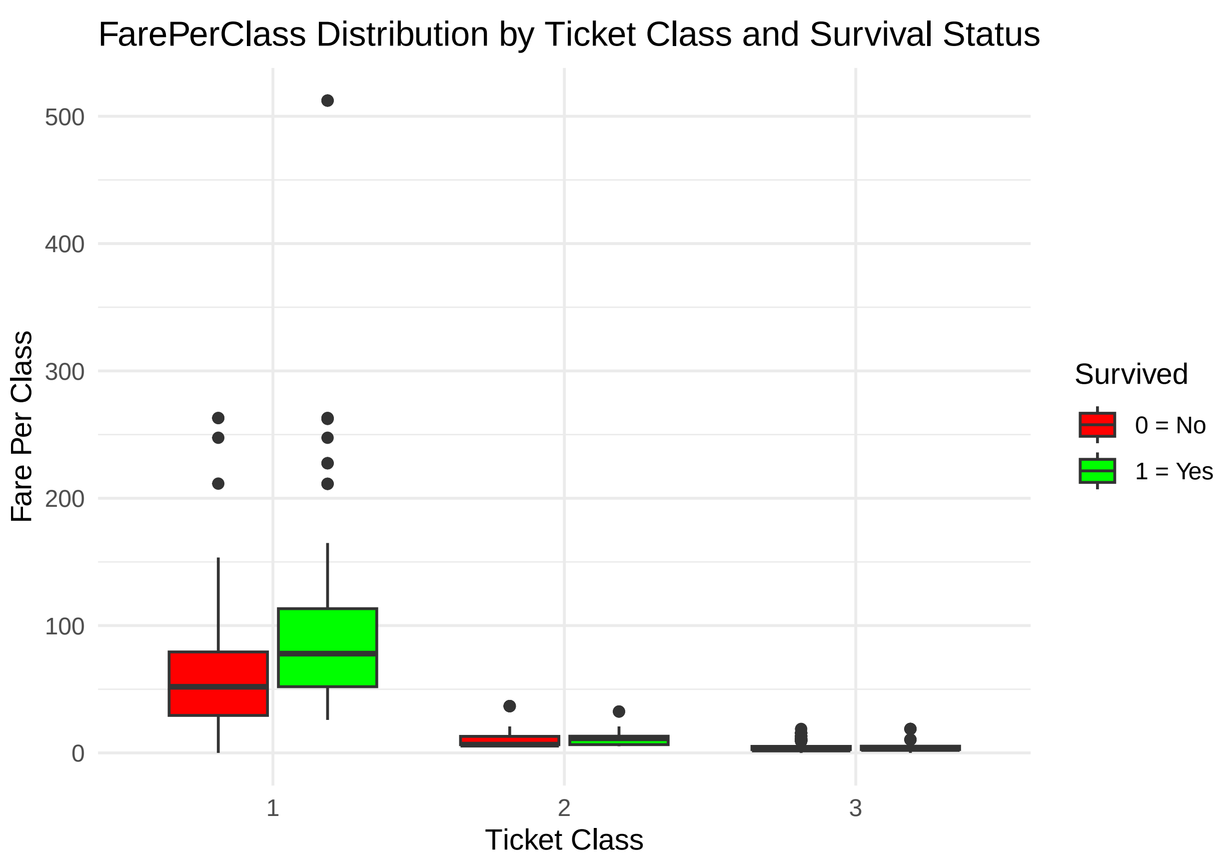
Observations:

1. Passengers in higher classes (lower Pclass value) tend to have higher FarePerClass values since they likely paid more for better accommodations.
2. For example:
   - Passenger 2 (First Class) paid 71.28, resulting in a FarePerClass of 71.28.

    - Passenger 1 (Third Class) paid 7.25, which becomes 2.42 after adjusting for the class.
3. This feature helps the model distinguish between passengers who paid high fares in different classes, which could be an important indicator for survival chances.

FarePerClass Plot

```r
# FarePerClass by Ticket Class and Survival Status
ggplot(train_df, aes(x = as.factor(Pclass), y = FarePerClass, fill = as.factor(Survived))) +
  geom_boxplot() +
  scale_fill_manual(values = c("red", "green"),
                    name = "Survived",
                    labels = c("0 = No", "1 = Yes")) +
  labs(
    title = "FarePerClass Distribution by Ticket Class and Survival Status",
    x = "Ticket Class",
    y = "Fare Per Class"
  ) +
  theme_minimal()
```



Inference:

- A clear positive relationship exists between FarePerClass and survival chances, particularly for first-class passengers.
- Passengers who paid higher fares within their class likely had better access to safety measures, such as quicker access to lifeboats or priority treatment during evacuation.

- The impact of FarePerClass on survival becomes less evident in lower ticket classes, possibly due to limited access to safety resources regardless of fare amount.

Categorizing Fare into Groups

```
# fare categories based on quartiles
train_df$FareCategory <- cut(
  train_df$Fare,
  breaks = c(-Inf, 7.91, 14.45, 31, Inf),
  labels = c("Low", "Medium", "High", "Very High")
)

test_df$FareCategory <- cut(
  test_df$Fare,
  breaks = c(-Inf, 7.91, 14.45, 31, Inf),
  labels = c("Low", "Medium", "High", "Very High")
)

# factor for modeling
train_df$FareCategory <- as.factor(train_df$FareCategory)
test_df$FareCategory <- as.factor(test_df$FareCategory)

head(train_df[, c("Fare", "FareCategory")])
```

```
##    Fare FareCategory
## 1  7.25          Low
## 2 71.28    Very High
## 3  7.92       Medium
## 4 53.10    Very High
## 5  8.05       Medium
## 6 51.86    Very High
```

The fares were divided into the following categories based on their value ranges:

- Low: Fares less than or equal to $7.91
- Medium: Fares between $7.91 and $14.45
- High: Fares between $14.45 and $31
- Very High: Fares greater than $31

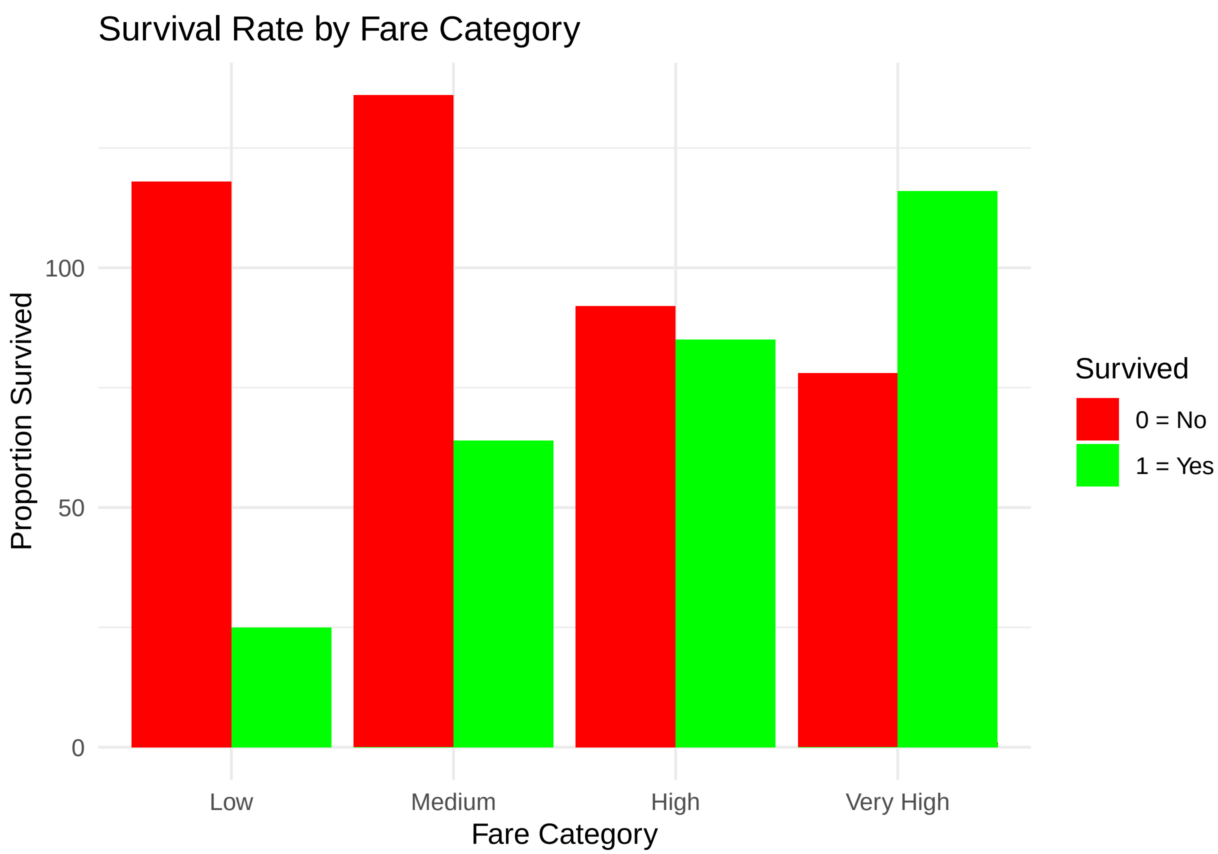
The output table displays the original Fare values alongside their corresponding FareCategory:

- A fare of the $7.25 was classified as Low.
- A fare of $71.28 was categorized as Very High.
- A fare of $7.92 was grouped into the Medium category.

By converting continuous numerical values into categorical groups, this feature engineering step helps to capture non-linear relationships between fare amounts and survival outcomes. This can lead to improved model performance, especially for algorithms like neural networks that can benefit from categorical input variables.

Categorized Fare Groups Survival Rate

```r
ggplot(train_df, aes(x = FareCategory, fill = as.factor(Survived))) +
  geom_bar(position = "fill") + # Proportional stacked bar plot
  geom_bar(position = "dodge") +
  scale_fill_manual(
    values = c("red", "green"),
    name = "Survived",
    labels = c("0 = No", "1 = Yes")
  ) +
  labs(
    title = "Survival Rate by Fare Category",
    x = "Fare Category",
    y = "Proportion Survived"
  ) +
  theme_minimal()
```



Insights from the Plot:

1. Low Fare Category:
   - The majority of passengers in this category did not survive.
   - This suggests that passengers who paid the lowest fares faced the highest risk of not surviving.
2. Medium Fare Category:
   - While the survival rate improved compared to the low fare group, more passengers still did not survive.
   - Passengers in this category had a moderately increased likelihood of survival.

3. High Fare Category:
    - Survival rates are closer to balanced in this category, with almost an equal proportion of survivors and non-survivors.
    - This indicates that higher fares were linked with better survival chances.
4. Very High Fare Category:
    - The majority of passengers in this category survived.
    - This strongly suggests that paying a very high fare increased the likelihood of survival, likely due to better access to lifeboats, priority treatment, and better cabin locations.

Inference:

- The plot demonstrates a clear positive correlation between fare category and survival probability: as the fare category increases, the proportion of passengers who survived also increases.
- This trend highlights the significant impact of socioeconomic status on survival rates during the Titanic disaster, as passengers who could afford higher fares likely had better access to safety measures.

Ticket Class as Categorical Variable

```r
# One-Hot encoding for Ticket Class (Pclass)
# converts Pclass into binary columns for each class

# convert Pclass to a factor to prepare for one-hot encoding
train_df$Pclass <- as.factor(train_df$Pclass)

# apply one-hot encoding using model.matrix
ticket_class_dummies <- model.matrix(~ Pclass - 1, data = train_df)

# combine the dummy variables with the original dataset
train_df <- cbind(train_df, ticket_class_dummies)

head(train_df)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           7        0      1
##                                                  Name Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   0  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   1  38     1     0
## 3                              Heikkinen, Miss. Laina   1  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel)   1  35     1     0
## 5                            Allen, Mr. William Henry   0  35     0     0
## 6                              McCarthy, Mr. Timothy J   0  54     0     0
##             Ticket  Fare Cabin Embarked Fare_Category FarePerClass FareCategory
## 1        A/5 21171  7.25              2           Low     2.416667          Low
## 2         PC 17599 71.28   C85        0          High    71.280000    Very High
## 3 STON/O2. 3101282  7.92              2           Low     2.640000       Medium
## 4           113803 53.10  C123        2          High    53.100000    Very High
## 5           373450  8.05              2           Low     2.683333       Medium
```

```
## 6          17463 51.86   E46        2          High    51.860000    Very High
##   Pclass1 Pclass2 Pclass3
## 1       0       0       1
## 2       1       0       0
## 3       0       0       1
## 4       1       0       0
## 5       0       0       1
## 6       1       0       0
```

Applying one-hot encoding to the Pclass variable, which represents the passenger ticket class (1st, 2nd, and 3rd class). Since machine learning models like neural networks require numerical input, categorical variables must be converted into a numerical format. One-hot encoding transforms each category into separate binary columns.

Output Interpretation: The resulting dataset now contains three additional columns:

- Pclass1: A binary column where 1 indicates the passenger was in 1st class, and 0 otherwise.
- Pclass2: A binary column where 1 indicates the passenger was in 2nd class, and 0 otherwise.
- Pclass3: A binary column where 1 indicates the passenger was in 3rd class, and 0 otherwise.

For example:

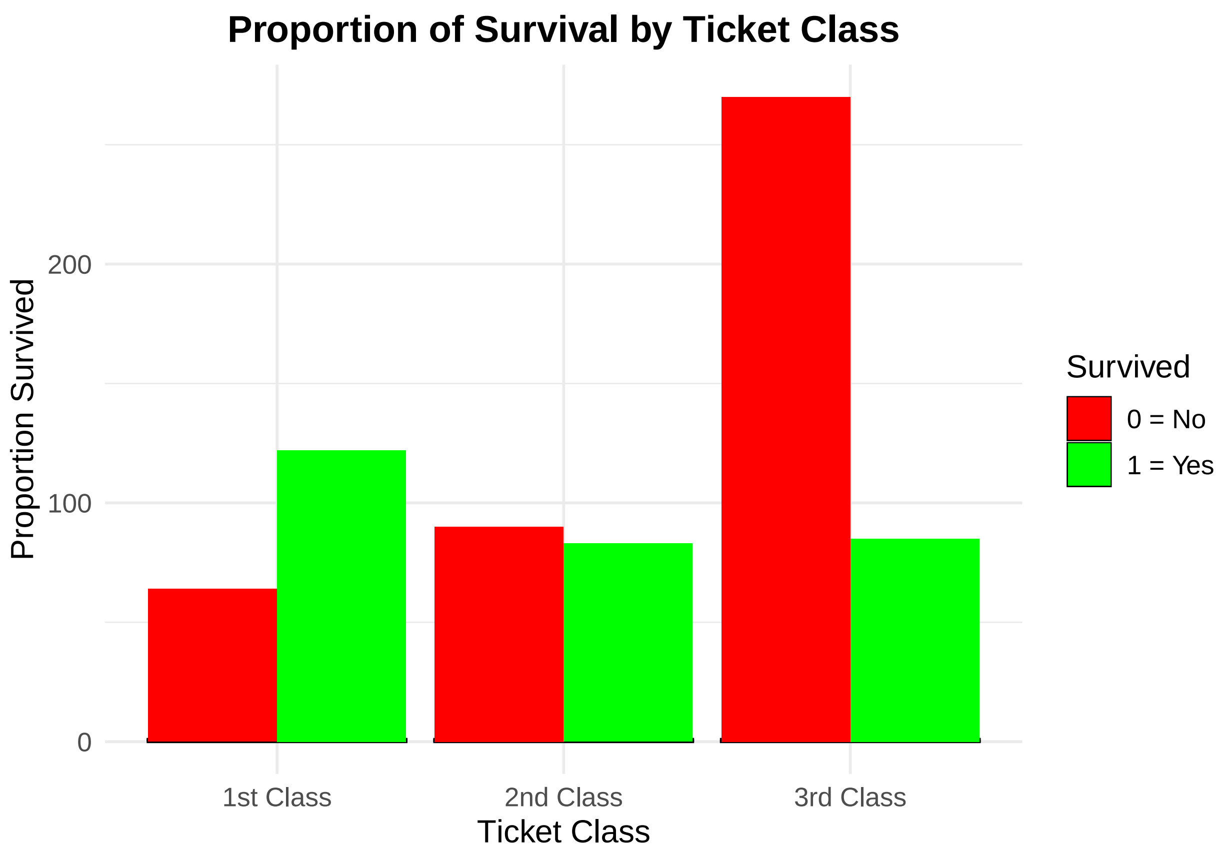
- The first passenger (Braund, Mr. Owen Harris) has Pclass3 = 1, indicating they were in the 3rd class.
- The second passenger (Cumings, Mrs. John Bradley) has Pclass1 = 1, indicating they were in the 1st class.

The transformation allows the ML model to process ticket class information numerically without implying any ordinal relationship between the classes.

Categorized Fare Groups Survival Rate

```r
train_df$TicketClass <- ifelse(train_df$Pclass1 == 1, "1st Class",
                         ifelse(train_df$Pclass2 == 1, "2nd Class", "3rd Class"))

ggplot(train_df, aes(x = TicketClass, fill = as.factor(Survived))) +
  geom_bar(position = "fill", color = "black") +
  geom_bar(position = "dodge") +
  scale_fill_manual(values = c("red", "green"), labels = c("0 = No", "1 = Yes"), name = "Survived") +
  labs(
    title = "Proportion of Survival by Ticket Class",
    x = "Ticket Class",
    y = "Proportion Survived"
  ) +
  theme_minimal() +
  theme(
    text = element_text(size = 12),
    plot.title = element_text(face = "bold", hjust = 0.5)
  )
```



Log Transformation of Fare

```r
# log Transformation of Fare to handle outliers
train_df$LogFare <- log(train_df$Fare + 1)
head(train_df[, c("Fare", "LogFare")])
```

```
##    Fare  LogFare
## 1  7.25 2.110213
## 2 71.28 4.280547
## 3  7.92 2.188296
## 4 53.10 3.990834
## 5  8.05 2.202765
## 6 51.86 3.967647
```

Insights from the Transformation:

1. LogFare Feature Creation:

- A new feature, LogFare, was created by applying a logarithmic transformation to the original Fare values.
- The transformation is applied using the formula:

$$LogFare = log(Fare + 1)$$

2. Impact of Transformation

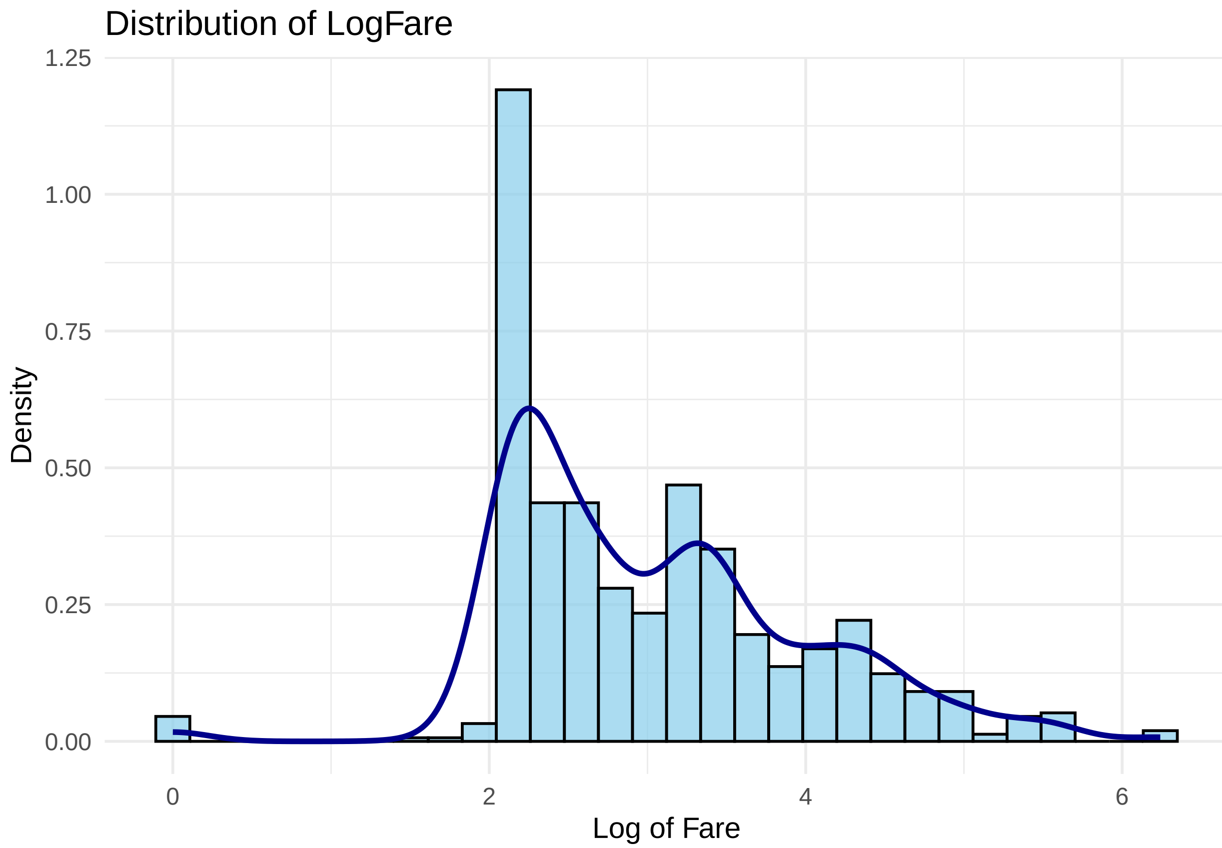
- Compresses the range of fare values, reducing the influence of extremely high fares.
- Helps in handling skewed data by transforming the fare distribution closer to normal.
- Ensures that outliers do not disproportionately affect the performance of the machine learning model.

LogFare Distribution Plot (Histogram + Density Plot)

```r
ggplot(train_df, aes(x = LogFare)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "skyblue", color = "black", alpha = 0.7) +
  geom_density(color = "darkblue", size = 1) +
  labs(
    title = "Distribution of LogFare",
    x = "Log of Fare",
    y = "Density"
  ) +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Inference: - Applying a logarithmic transformation helped to normalize the originally skewed fare distribution. - The plot suggests that while most passengers paid relatively lower fares, there are distinct groups of passengers who paid higher fares. - Normalizing the fare values through log transformation will likely improve the model's performance, especially when training the Neural Network (MLPClassifier), as it stabilizes variance and makes patterns in the data clearer.

Heatmap

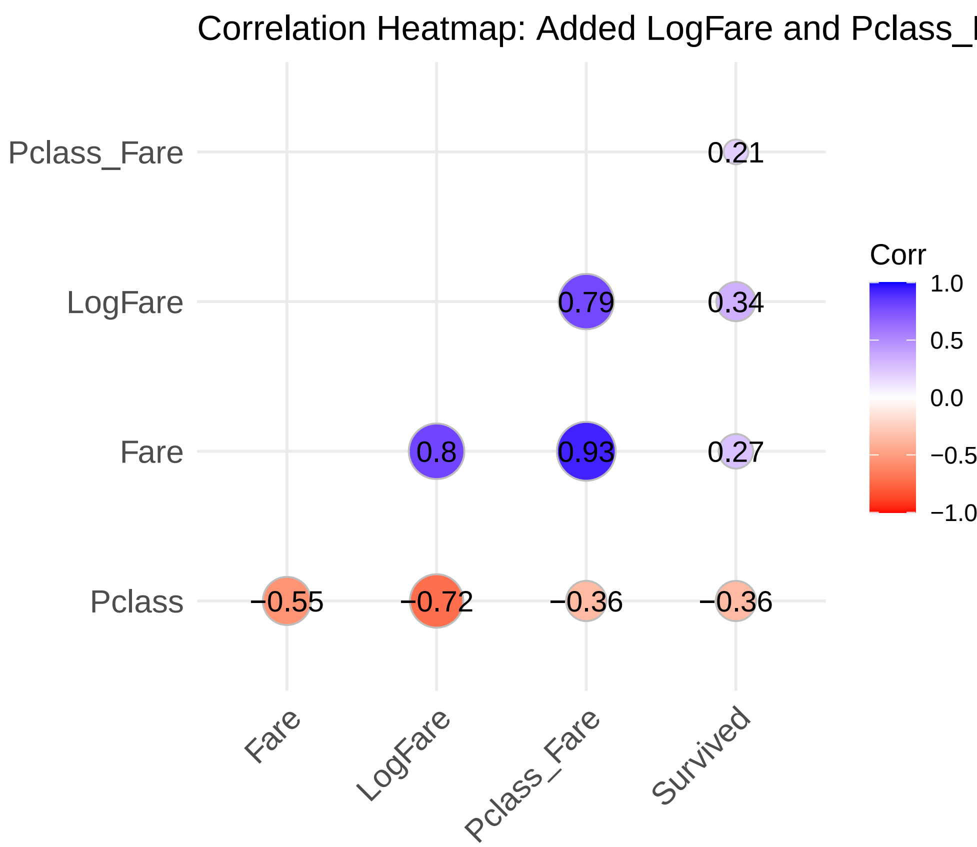
```
#install.packages("ggcorrplot")
library(ggcorrplot)

correlation_features <- train_df %>%
  mutate(
    Pclass = as.numeric(Pclass),
    Fare = as.numeric(Fare),
    Survived = as.numeric(Survived),
    LogFare = log1p(Fare),
    Pclass_Fare = Pclass * Fare
  ) %>%
  select(Pclass, Fare, LogFare, Pclass_Fare, Survived)

correlation_matrix <- round(cor(correlation_features, use = "complete.obs"), 2)
```

```r
ggcorrplot(
  correlation_matrix,
  method = "circle",
  type = "lower",
  lab = TRUE,
  colors = c("red", "white", "blue"),
  title = "Correlation Heatmap: Added LogFare and Pclass_Fare",

)
```



Observations:

1. Pclass (Passenger Class) Correlations:
    - Pclass vs. Fare (-0.55): A moderate negative correlation. Higher-class passengers (lower Pclass values) tend to have higher fares.
    - Pclass vs. LogFare (-0.72): A strong negative correlation. This is expected since LogFare is a transformation of Fare.
    - Pclass vs. Survived (-0.36): Negative correlation indicates passengers in higher classes were more likely to survive.
2. Fare Correlations:
    - Fare vs. LogFare (0.80): A strong positive correlation since LogFare is just a log-transformation of Fare.
    - Fare vs. Pclass_Fare (0.93): Extremely strong positive correlation because Pclass_Fare is derived by multiplying Pclass and Fare.

- Fare vs. Survived (0.27): A weak-to-moderate positive correlation implies passengers who paid higher fares were more likely to survive.

3. LogFare Correlations:

    - LogFare vs. Survived (0.34): Moderate positive correlation suggests that higher logged fares relate to better survival odds.

4. Pclass_Fare (Interaction Term):

    - Pclass_Fare vs. Survived (0.21): Weak positive correlation, indicating a slight influence on survival based on class-fare interaction.

Inference:

- The most influential features for survival based on the correlations are Fare, Pclass, and LogFare. This suggests that socio-economic factors played a significant role in determining survival odds during the Titanic disaster.

Acuracy

```
#install.packages("scales")

library(scales)  # For normalization
```

```
##
## Attaching package: 'scales'

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

## The following object is masked from 'package:readr':
##
##     col_factor
```

```
train_df <- read.csv("clean_train.csv", stringsAsFactors = FALSE)

train_df <- train_df %>%
  mutate(
    Pclass = as.numeric(Pclass),
    Fare = as.numeric(Fare),
    Age = as.numeric(Age),
    Sex = as.numeric(Sex),
    Embarked = as.numeric(Embarked),
    LogFare =log1p(Fare),
    Pclass_Fare = Pclass * Fare,
    Survived = as.factor(Survived)
  ) %>%
  select(Survived, Pclass, Fare, Age, Sex, Embarked, LogFare, Pclass_Fare)

# normalize continuous features (Fare and Age)
train_df$Fare <- rescale(train_df$Fare)
train_df$Age <- rescale(train_df$Age)
train_df$LogFare <- rescale(train_df$LogFare)
```

```r
# split into training and testing sets
set.seed(123)
n <- nrow(train_df)
train_index <- sample(1:n, size = 0.8 * n)
train_set <- train_df[train_index, ]
test_set <- train_df[-train_index, ]

# train MLPClassifier
mlp_model <- nnet(Survived ~ Pclass + Fare + Age + Sex + Embarked + LogFare + Pclass_Fare,
                  data = train_set,
                  size = 30,
                  decay = 0.0005,
                  maxit = 2000,
                  trace = FALSE)

#predictions
predictions <- predict(mlp_model, test_set, type = "class")

actual <- test_set$Survived
correct_predictions <- sum(predictions == actual)
total_predictions <- length(actual)
accuracy <- correct_predictions / total_predictions

cat("\n Enhanced MLP Model Performance \n")
```

```
##
##  Enhanced MLP Model Performance
```

```r
cat(sprintf("Accuracy: %.2f%%\n", accuracy * 100))
```

```
## Accuracy: 82.52%
```

```r
# confusion matrix
conf_matrix <- table(Predicted = predictions, Actual = actual)
cat("\nConfusion Matrix:\n")
```

```
##
## Confusion Matrix:
```

```r
print(conf_matrix)
```

```
##          Actual
## Predicted  0  1
##         0 71 17
##         1  8 47
```

```r
# Precision, Recall, and F1 Score
TP <- conf_matrix["1", "1"]
TN <- conf_matrix["0", "0"]
FP <- conf_matrix["1", "0"]
FN <- conf_matrix["0", "1"]

precision <- TP / (TP + FP)
recall <- TP / (TP + FN)
f1_score <- 2 * ((precision * recall) / (precision + recall))
```

```r
cat(sprintf("\nPrecision: %.2f", precision))
```

```
##
## Precision: 0.85
```

```r
cat(sprintf("\nRecall: %.2f", recall))
```

```
##
## Recall: 0.73
```

```r
cat(sprintf("\nF1 Score: %.2f\n", f1_score))
```

```
##
## F1 Score: 0.79
```

Model Accuracy

Accuracy: 82.52% - The model predicted survival for approximately 82.52% of the passengers in the test set.

- Formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Confusion Matrix

	Actual Survived (1)	Actual Not Survived (0)
Predicted 1	47 (True Positives)	8 (False Positives)
Predicted 0	17 (False Negatives)	71 (True Negatives)

- True Positives (TP) = 47
  - The model correctly predicted survival.
- True Negatives (TN) = 71
  - The model correctly predicted non-survival.
- False Positives (FP) = 8
  - The model incorrectly predicted survival when the passenger didn't survive.
- False Negatives (FN) = 17 The model incorrectly predicted non-survival when the passenger actually survived.

Performance Metrics

- Precision: 0.85
  - Out of all passengers predicted to survive, 85% actually survived.
  - Formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Recall: 0.73
  - Out of all actual survivors, the model correctly identified 73% of them.

  - Formula:

$$\text{Recall} = \frac{TP}{TP + FN}$$

- F1 Score: 0.79
  - The harmonic mean of precision and recall, balancing the two.
  - Formula:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$
