Part 1 - Exploration

Ibrahim Yazici

Load packages

We use the `tidyverse` suite of packages.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Read data

The code chunk below reads in the final project data.

```
df <- readr::read_csv("paint_project_train_data.csv", col_names = TRUE)
```

```
## Rows: 835 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): Lightness, Saturation
## dbl (6): R, G, B, Hue, response, outcome
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The `readr::read_csv()` function displays the data types and column names associated with the data. However, a glimpse is shown below that reveals the number of rows and also shows some of the representative values for the columns.

```
df %>% glimpse()
```

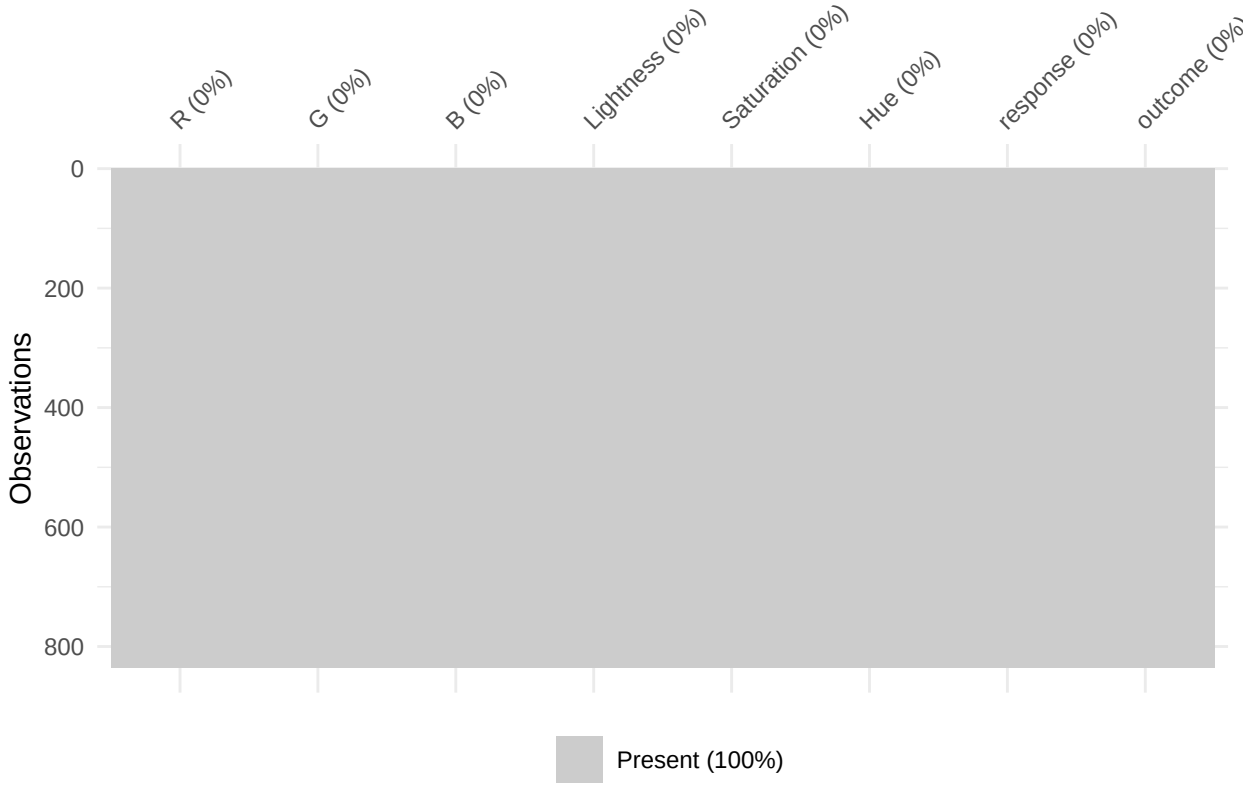
The data consist of continuous and categorical inputs. The `glimpse()` shown above reveals the data type for each variable which state to us whether the input is continuous or categorical. The RGB color model inputs, `R`, `G`, and `B` are continuous (dbl) inputs. The HSL color model inputs consist of 2 categorical inputs, `Lightness` and `Saturation`, and a continuous input, `Hue`. Two outputs are provided. The continuous output, `response`, and the Binary output, `outcome`. However, the data type of the Binary outcome is numeric because the Binary `outcome` is **encoded** as `outcome = 1` for the EVENT and `outcome = 0` for the NON-EVENT.

Exploration

Basic information

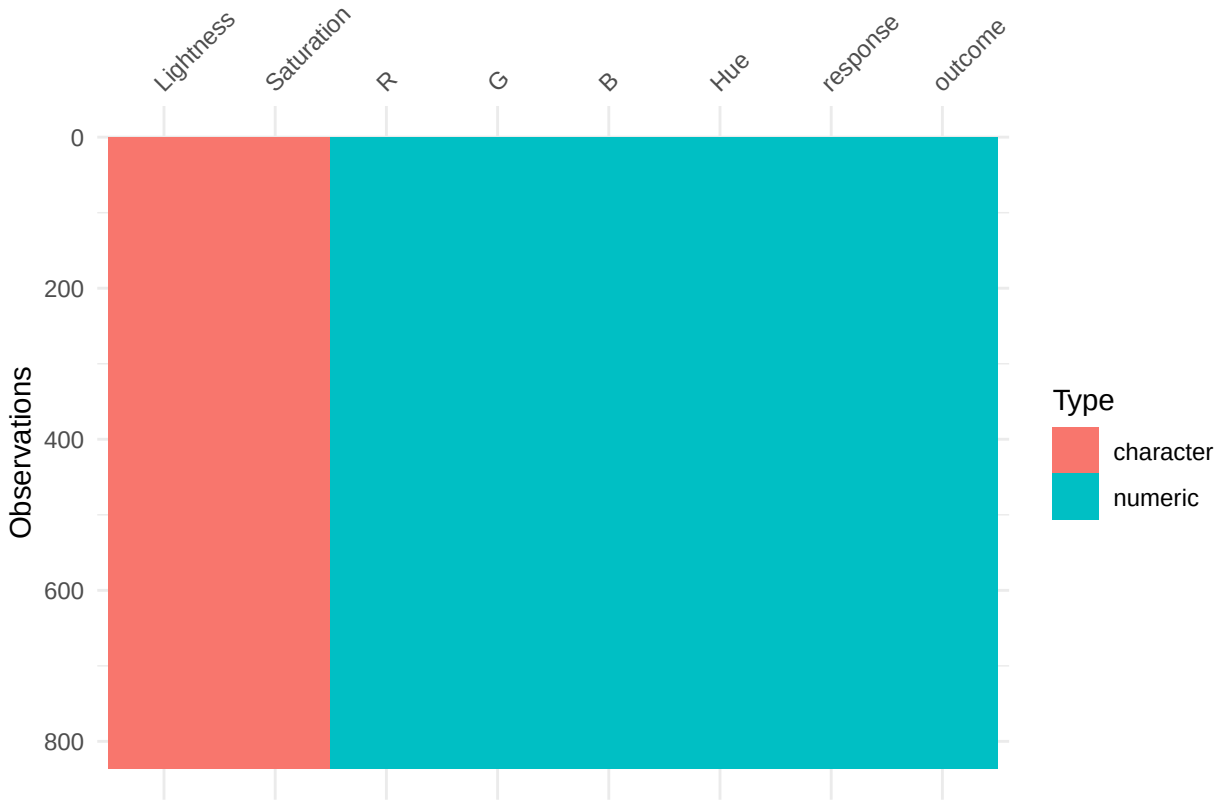
Visually check for missing values using the visdat package.

```
visdat::vis_miss(df)
```



Check the data types visually.

```
visdat::vis_dat(df)
```



Check the number of unique values per variable.

```
df %>% purrr::map_dbl(n_distinct)
```

```
##          R          G          B  Lightness Saturation        Hue   response
##        197        171        183          7          7         36         83
##    outcome
##          2
```

a) Visualize the distributions of variables in the data set.

The following plots show counts for cathegorical variables.

```
ggplot(df, aes(x = Lightness)) +
  geom_bar() +
  theme_minimal() +
  labs(title = "Count of Lightness Categories", x = "Lightness", y = "Count")

ggplot(df, aes(x = Saturation)) +
  geom_bar() +
  theme_minimal() +
  labs(title = "Count of Saturation Categories", x = "Saturation", y = "Count")
```

The following are Histograms plots for continuous variables.

```
ggplot(df, aes(x = R)) +
  geom_histogram(bins = 30,fill ="blue", color = "black") +
  labs(title = "Histogram for R", x = "R", y = "Count") +
  theme_minimal()

ggplot(df, aes(x = G)) +
```

```
  geom_histogram(bins = 30,fill ="green", color = "black") +
  labs(title = "Histogram for G", x = "G", y = "Count") +
  theme_minimal()

ggplot(df, aes(x = B)) +
  geom_histogram(bins = 30,fill ="red", color = "black") +
  labs(title = "Histogram for B", x = "B", y = "Count") +
  theme_minimal()

ggplot(df, aes(x = Hue)) +
  geom_histogram(bins = 30,fill ="orange", color = "black") +
  labs(title = "Histogram for Hue", x = "Hue", y = "Count") +
  theme_minimal()
```

The following are Density plots for continuous variables.

```
ggplot(df, aes(x = R)) +
  geom_density(fill ="blue") +
  labs(title = "Density Plot for R", x = "R", y = "Density") +
  theme_minimal()

ggplot(df, aes(x = G)) +
  geom_density(fill ="green") +
  labs(title = "Density Plot for G", x = "G", y = "Density") +
  theme_minimal()

ggplot(df, aes(x = B)) +
  geom_density(fill ="red") +
  labs(title = "Density Plot for B", x = "B", y = "Density") +
  theme_minimal()

ggplot(df, aes(x = Hue)) +
  geom_density(fill ="orange") +
  labs(title = "Density Plot for Hue", x = "Hue", y = "Density") +
  theme_minimal()
```

To determine if the distributions are Gaussian-like, since we cannot visually observe the symmetry and bell shape of the histograms or density plots above, we conclude that the distributions are not Gaussian-like. We can also compute skewness values for the continuous variables:

```
library(e1071)
skewness_R <- skewness(df$R)
skewness_R
skewness_G <- skewness(df$G)
skewness_G
skewness_B <- skewness(df$B)
skewness_B
skewness_Hue <- skewness(df$Hue)
skewness_Hue
```

b) Condition (group) the continuous variables based on the categorical variables.

The following plots provide Histogram of R conditioned on the cathegorical variables. After each plot a table which calculates summary statistics based on categorical variable values is given.

```
ggplot(df, aes(x = R, fill = Lightness)) +
  geom_histogram(bins= 30) +
  labs(title="Histogram of R conditioned on Lightness", x = "R", y = "Count")

df %>%
  group_by(Lightness) %>%
  summarise(Mean_R = mean(R), SD_R = sd(R), Min_R = min(R), Max_R = max(R))

ggplot(df, aes(x = R, fill = Saturation)) +
  geom_histogram(bins= 30) +
  labs(title ="Histogram of R conditioned on Saturation", x = "R", y = "Count")

df %>%
  group_by(Saturation) %>%
  summarise(Mean_R = mean(R), SD_R = sd(R), Min_R = min(R), Max_R = max(R))
```

The following plots provide Histogram of G conditioned on the cathegorical variables. After each plot a table which calculates summary statistics based on categorical variable values is given.

```
ggplot(df, aes(x = G, fill = Lightness)) +
  geom_histogram(bins= 30) +
  labs(title="Histogram of G conditioned on Lightness", x = "G", y = "Count")

df %>%
  group_by(Lightness) %>%
  summarise(Mean_G = mean(G), SD_G = sd(G), Min_G = min(G), Max_G = max(G))

ggplot(df, aes(x = G, fill = Saturation)) +
  geom_histogram(bins= 30) +
  labs(title ="Histogram of G conditioned on Saturation", x = "G", y = "Count")

df %>%
  group_by(Saturation) %>%
  summarise(Mean_G = mean(G), SD_G = sd(G), Min_G = min(G), Max_G = max(G))
```

The following plots provide Histogram of B conditioned on the cathegorical variables. After each plot a table which calculates summary statistics based on categorical variable values is given.

```
ggplot(df, aes(x = B, fill = Lightness)) +
  geom_histogram(bins= 30) +
  labs(title="Histogram of B conditioned on Lightness", x = "B", y = "Count")

df %>%
  group_by(Lightness) %>%
  summarise(Mean_B = mean(B), SD_B = sd(B), Min_B = min(B), Max_B = max(B))

ggplot(df, aes(x = B, fill = Saturation)) +
  geom_histogram(bins= 30) +
  labs(title ="Histogram of B conditioned on Saturation", x = "B", y = "Count")

df %>%
  group_by(Saturation) %>%
  summarise(Mean_B = mean(B), SD_B = sd(B), Min_B = min(B), Max_B = max(B))
```

The following plots provide Histogram of Hue conditioned on the cathegorical variables. After each plot a

table which calculates summary statistics based on categorical variable values is given.

```
ggplot(df, aes(x = Hue, fill = Lightness)) +
  geom_histogram(bins= 30) +
  labs(title="Histogram of Hue conditioned on Lightness", x = "Hue", y = "Count")

df %>%
  group_by(Lightness) %>%
  summarise(Mean_Hue = mean(Hue), SD_Hue = sd(Hue), Min_Hue = min(Hue), Max_Hue = max(Hue))

ggplot(df, aes(x = Hue, fill = Saturation)) +
  geom_histogram(bins= 30) +
  labs(title ="Histogram of Hue conditioned on Saturation", x = "Hue", y = "Count")

df %>%
  group_by(Saturation) %>%
  summarise(Mean_Hue = mean(Hue), SD_Hue = sd(Hue), Min_Hue = min(Hue), Max_Hue = max(Hue))
```

c) Condition (group) the continuous variables based on the binary outcome.

The following plot provides Histogram of R conditioned on the binary outcome. After the plot a table which calculates summary statistics based on the binary outcome is given.

```
ggplot(df, aes(x = R, fill = factor(outcome))) +
  geom_histogram(bins = 30) +
  labs(title="Histogram of R conditioned on Outcome", x = "R", y = "Count")

df %>%
  group_by(outcome) %>%
  summarise(Mean_R = mean(R), SD_R = sd(R), Min_R = min(R), Max_R = max(R))
```

The following plot provides Histogram of G conditioned on the binary outcome. After the plot a table which calculates summary statistics based on the binary outcome is given.

```
ggplot(df, aes(x = G, fill = factor(outcome))) +
  geom_histogram(bins = 30) +
  labs(title="Histogram of G conditioned on Outcome", x = "G", y = "Count")

df %>%
  group_by(outcome) %>%
  summarise(Mean_G = mean(G), SD_G = sd(G), Min_G = min(G), Max_G = max(G))
```

The following plot provides Histogram of B conditioned on the binary outcome. After the plot a table which calculates summary statistics based on the binary outcome is given.

```
ggplot(df, aes(x = B, fill = factor(outcome))) +
  geom_histogram(bins = 30) +
  labs(title="Histogram of B conditioned on Outcome", x = "B", y = "Count")

df %>%
  group_by(outcome) %>%
  summarise(Mean_B = mean(B), SD_B = sd(B), Min_B = min(B), Max_B = max(B))
```

The following plot provides Histogram of Hue conditioned on the binary outcome. After the plot a table which calculates summary statistics based on the binary outcome is given.

```r
ggplot(df, aes(x = Hue, fill = factor(outcome))) +
  geom_histogram(bins = 30) +
  labs(title="Histogram of Hue conditioned on Outcome", x = "Hue", y = "Count")

df %>%
  group_by(outcome) %>%
  summarise(Mean_Hue = mean(Hue), SD_Hue = sd(Hue), Min_Hue = min(Hue), Max_Hue = max(Hue))
```

d) Visualizing the relationships between the continuous inputs.

The following plots help us visualize the relationships between the continuous inputs.

```r
ggplot(df, aes(x=R, y=G)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of R vs G", x = "R", y = "G") +
  theme_minimal()

ggplot(df, aes(x=R, y=B)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of R vs B", x = "R", y = "B") +
  theme_minimal()

ggplot(df, aes(x=R, y=Hue)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of R vs Hue", x = "R", y = "Hue") +
  theme_minimal()

ggplot(df, aes(x=G, y=B)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of G vs B", x = "G", y = "B") +
  theme_minimal()

ggplot(df, aes(x=G, y=Hue)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of G vs Hue", x = "G", y = "Hue") +
  theme_minimal()

ggplot(df, aes(x=B, y=Hue)) +
  geom_point() +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Scatter Plot of B vs Hue", x = "B", y = "Hue") +
  theme_minimal()
```

Below, we can see the correlation coefficients between the continuous inputs.

```r
cor(df$R, df$G)
cor(df$R, df$B)
cor(df$R, df$Hue)
cor(df$G, df$B)
cor(df$G, df$Hue)
```

```
cor(df$B, df$Hue)
```

e) Visualizing the relationships between the continuous outputs (response and the LOGIT transformed response) with respect to the continuous inputs.

The plots below are given to visualize the relationships between the continuous outputs (response and the LOGIT transformed response) with respect to the continuous inputs `R`. They use color to differentiate `Lightness` and facets for `Saturation`.

```
dff <- df %>%
  mutate(y = boot::logit((response - 0) / (100 - 0)))

ggplot(dff, aes(x=R, y= response, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Response vs R", x = "R", y = "Response") +
  theme_minimal()

ggplot(dff, aes(x=R, y= y, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Logit(Response) vs R", x = "R", y = "Logit(Response)") +
  theme_minimal()
```

The plots below are given to visualize the relationships between the continuous outputs (response and the LOGIT transformed response) with respect to the continuous inputs `G`. They use color to differentiate `Lightness` and facets for `Saturation`.

```
ggplot(dff, aes(x=G, y= response, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Response vs G", x = "G", y = "Response") +
  theme_minimal()

ggplot(dff, aes(x=G, y= y, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Logit(Response) vs G", x = "G", y = "Logit(Response)") +
  theme_minimal()
```

The plots below are given to visualize the relationships between the continuous outputs (response and the LOGIT transformed response) with respect to the continuous inputs `B`. They use color to differentiate `Lightness` and facets for `Saturation`.

```
ggplot(dff, aes(x=B, y= response, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Response vs B", x = "B", y = "Response") +
  theme_minimal()

ggplot(dff, aes(x=B, y= y, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Logit(Response) vs B", x = "B", y = "Logit(Response)") +
  theme_minimal()
```

The plots below are given to visualize the relationships between the continuous outputs (response and the LOGIT transformed response) with respect to the continuous inputs `Hue`. They use color to differentiate `Lightness` and facets for `Saturation`.

```
ggplot(dff, aes(x=Hue, y= response, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Response vs Hue", x = "Hue", y = "Response") +
  theme_minimal()

ggplot(dff, aes(x=Hue, y= y, color = Lightness)) +
  geom_point() +
  facet_wrap(~Saturation) +
  labs(title = "Logit(Response) vs Hue", x = "Hue", y = "Logit(Response)") +
  theme_minimal()
```

f) Visualizing the behavior of the binary outcome.

The boxplots below are given to visualize the relationships between the binary outcome and continuous input R.

```
ggplot(dff, aes(x = as.factor(outcome), y = R)) +
  geom_boxplot() +
  labs(title= "Boxplot of R for Each Outcome", x = "Outcome", y = "R") +
  theme_minimal()
```

The boxplots below are given to visualize the relationships between the binary outcome and continuous input G.

```
ggplot(dff, aes(x = as.factor(outcome), y = G)) +
  geom_boxplot() +
  labs(title= "Boxplot of G for Each Outcome", x = "Outcome", y = "G") +
  theme_minimal()
```

The boxplots below are given to visualize the relationships between the binary outcome and continuous input B.

```
ggplot(dff, aes(x = as.factor(outcome), y = B)) +
  geom_boxplot() +
  labs(title= "Boxplot of B for Each Outcome", x = "Outcome", y = "B") +
  theme_minimal()
```

The boxplots below are given to visualize the relationships between the binary outcome and continuous input Hue.

```
ggplot(dff, aes(x = as.factor(outcome), y = Hue)) +
  geom_boxplot() +
  labs(title= "Boxplot of Hue for Each Outcome", x = "Outcome", y = "Hue") +
  theme_minimal()
```

The bar plot below is given to visualize the relationships between the binary outcome and categorical input Lightness.

```
ggplot(dff, aes(x = Lightness, fill = as.factor(outcome))) +
  geom_bar() +
  labs(title= "Binary Outcome by Lightness", x = "Lightness", y = "Count(Outcome)") +
  theme_minimal()
```

The bar plot below is given to visualize the relationships between the binary outcome and categorical input Saturation.

```
ggplot(dff, aes(x = Saturation, fill = as.factor(outcome))) +
  geom_bar() +
  labs(title= "Binary Outcome by Saturation", x = "Saturation", y = "Count(Outcome)") +
  theme_minimal()
```