Appendix

Data Cleaning and Reformating

```
if (!require("olsrr")) {
  install.packages("olsrr")
,eval = FALSE}
library(olsrr)

#Data cleaning
library(readr)
ted_data <- read_csv("ted_main.csv", locale = locale(encoding = "UTF-8"))
ted_data <- ted_data[complete.cases(ted_data[, c(
  "comments",
  "duration",
  "languages",
  "num_speaker",
  "published_date",
  "ratings",
  "tags",
  "views"
)]), ]

ted_data$film_date <- as.POSIXct(ted_data$film_date, origin = "1970-01-01", tz = "UTC")
ted_data$published_date <- as.POSIXct(ted_data$published_date, origin = "1970-01-01", tz = "UTC")

library(dplyr)
library(tidyverse)
library(car)
library(olsrr)
current_date <- as.Date("2025-04-04")
ted_data$published_date <- as.Date(ted_data$published_date)
ted_data$video_age <- as.numeric(current_date - ted_data$published_date)
ted_data<- ted_data %>% mutate(views_per_day = views/video_age)
ted_data <- ted_data %>% relocate(video_age, views_per_day, .after = published_date)

## this can be negative
ted_data <- ted_data %>%
  mutate(popularity_score = log1p(views_per_day) +
                            log1p(comments))


library(stringr)
library(dplyr)
ted_data$tag_count <- str_count(ted_data$tags, "'[^']+'")
ted_comments <- ted_data %>% arrange(desc(comments))
ted_views <- ted_data %>% arrange(desc(views))
ted_tag_count <- ted_data %>% arrange(desc(tag_count))
ted_duration <- ted_data %>% arrange(desc(duration))
ted_languages <- ted_data %>% arrange(desc(languages))
ted_film_date <- ted_data %>% arrange(desc(film_date))
ted_publish_date <- ted_data %>% arrange(desc(published_date))
```

```r
#Goal for this part of code:
#1. Normalize each video's ratings (as proportions)
#2. Identify the top 3 most common rating types across all videos
#3. Flag whether each video includes those ratings in its top 5 most frequent ratings


# Check Ratings
library(dplyr)
library(tidyr)
library(jsonlite)
library(stringr)
library(purrr)
# Step 1: Fix JSON formatting and parse ratings
ted_data <- ted_data %>%
  mutate(ratings_fixed = str_replace_all(ratings, "'", "\""),
         video_id = row_number()) %>%
  mutate(ratings_parsed = map(ratings_fixed, ~fromJSON(.x)))

# Step 2: Unnest ratings into long format
ratings_long <- ted_data %>%
  dplyr::select(video_id, ratings_parsed) %>%
  unnest(ratings_parsed)  # columns: video_id, id, name, count

# Step 3: Normalize counts to percentages within each video
ratings_long <- ratings_long %>%
  group_by(video_id) %>%
  mutate(pct = count / sum(count)) %>%
  ungroup()

# Step 4: Compute overall average percentage for each rating
rating_summary <- ratings_long %>%
  group_by(name) %>%
  summarise(mean_pct = mean(pct, na.rm = TRUE)) %>%
  arrange(desc(mean_pct))

# Step 5: Select top 3 most common ratings across all videos
top3_ratings <- rating_summary %>%
  slice_head(n = 3) %>%
  pull(name)

# Step 6: Identify top 5 ratings (by count) within each video
top5_per_video <- ratings_long %>%
  group_by(video_id) %>%
  slice_max(order_by = count, n = 5) %>%
  ungroup()

# Step 7: For each of the top 3 ratings, create indicator columns
rating_flags <- top5_per_video %>%
  filter(name %in% top3_ratings) %>%
  mutate(flag = 1) %>%
  pivot_wider(names_from = name,
              values_from = flag,
              values_fill = 0,
```

```
              names_prefix = "Is_rating_t5_")

# Step 8: Merge flags back into original dataset
ted_data <- ted_data %>%
  left_join(rating_flags, by = "video_id") %>%
  mutate(across(starts_with("Is_rating_t5_"), ~replace_na(.x, 0)))

#Find the top 10 of the tags
ted_data <- ted_data %>% relocate(tag_count, .after = tags)
all_tags <- unlist(str_extract_all(ted_data$tags, "'[^']+'"))
all_tags <- str_replace_all(all_tags, "'", "")
tag_freq <- table(all_tags)
tag_freq_df <- as.data.frame(tag_freq) %>%
  arrange(desc(Freq))

# Add new tag-checking columns to the dataset
library(stringr)
ted_data$tag_is_technology <- ifelse(str_detect(tolower(ted_data$tags), "technology"), 1, 0)
ted_data$tag_is_science <- ifelse(str_detect(tolower(ted_data$tags), "science"), 1, 0)
ted_data$tag_is_global_issues <- ifelse(str_detect(tolower(ted_data$tags), "global issues"), 1, 0)

ted_data <- ted_data[, c("popularity_score", "duration", "languages", "num_speaker",
                                  "video_age", "tag_count", "Is_rating_t5_Inspiring",
                                  "tag_is_technology", "tag_is_science", "tag_is_global_issues")]
#names(ted_data)
write.csv(ted_data, file = "ted_data.csv", row.names = FALSE)
```

---

2. Exploratory Data Analysis

2.1 Summary of Variables

```
summary(ted_data)
```

2.2 Quantitative Variables

```
library(ggplot2)
library(GGally)
library(scales)

# Subset the numeric variables
numeric_vars <- ted_data[, c("popularity_score", "duration", "languages",
                             "num_speaker", "video_age", "tag_count")]

library(GGally)
library(ggplot2)
```

```r
upper_cor_colored <- function(data, mapping, digits = 2, ...) {
  # Extract the x and y variables
  x <- eval_data_col(data, mapping$x)
  y <- eval_data_col(data, mapping$y)

  # Compute correlation
  corr <- cor(x, y, use = "complete.obs")

  df <- data.frame(corr = corr, x = 1, y = 1)

  # Build a ggplot tile with text
  ggplot(df, aes(x, y, fill = corr)) +
    geom_tile() +
    # Show the correlation value in text
    geom_text(aes(label = round(corr, digits)), color = "black", size = 5) +
    # Color gradient: red (negative), white (zero), blue (positive)
    scale_fill_gradient2(
      low = "red", mid = "white", high = "blue",
      midpoint = 0, limits = c(-1, 1)
    ) +
    theme_void() +                  # Remove axes, etc.
    theme(legend.position = "none")  # Hide legend
,eval = FALSE}

# Example ggpairs call using the custom function in the 'upper' panels
p <- ggpairs(
  numeric_vars,
  title = "Pairs Plot of Numeric Variables (GGally)",
  lower = list(continuous = wrap("smooth", color = "red", se = FALSE)),
  diag  = list(continuous = wrap("barDiag", fill = "blue")),
  upper = list(continuous = wrap(upper_cor_colored))
)
```

2.3 Qualitative Variables

```r
# Set up the plotting area to have 1 row and 4 columns
par(mfrow = c(1, 4))

# Boxplot of popularity_score vs Is_rating_t5_Inspiring
ted_data$Is_rating_t5_Inspiring <- as.factor(ted_data$Is_rating_t5_Inspiring)
boxplot(popularity_score ~ Is_rating_t5_Inspiring, data = ted_data,
        main = "Popularity Score vs \n Inspiring Rating",
        xlab = "Is Rating T5 Inspiring", ylab = "Popularity Score",
        cex.main = 1.2,cex.lab = 1.2)

# Boxplot of popularity_score vs tag_is_technology
ted_data$tag_is_technology <- as.factor(ted_data$tag_is_technology)
boxplot(popularity_score ~ tag_is_technology, data = ted_data,
        main = "Popularity Score vs \n Technology Tag",
        xlab = "Tag is Technology", ylab = "Popularity Score",
        cex.main = 1.2,cex.lab = 1.2)
```

```
# Boxplot of popularity_score vs tag_is_science
ted_data$tag_is_science <- as.factor(ted_data$tag_is_science)
boxplot(popularity_score ~ tag_is_science, data = ted_data,
        main = "Popularity Score vs \n Science Tag",
        xlab = "Tag is Science", ylab = "Popularity Score",
        cex.main = 1.2,cex.lab = 1.2)

# Boxplot of popularity_score vs tag_is_global_issues
ted_data$tag_is_global_issues <- as.factor(ted_data$tag_is_global_issues)
boxplot(popularity_score ~ tag_is_global_issues, data = ted_data,
        main = "Popularity Score vs \n Global Issues Tag",
        xlab = "Tag is Global Issues", ylab = "Popularity Score",
        cex.main = 1.2,cex.lab = 1.2)
```

3: Fit Initial Full Model (Main Effects Only)

```
full_model <- lm( popularity_score ~ duration + languages + num_speaker + video_age + tag_count +  Is_ra
                    tag_is_science + tag_is_global_issues,data = ted_data
)
```

4 Check for Linearity, Curvature, and Interaction Effects

```
library(olsrr)
avPlots(full_model)
par(mfrow=c(1,3))

# install.packages("patchwork") # if not already installed
library(ggplot2)
library(patchwork)

plot1 <- ggplot(ted_data, aes(duration, popularity_score)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", formula = y ~ poly(x, 2), color = "red") +
  labs(title = "Curved Relationship between Duration and Popularity",
       x = "Duration (seconds)",
       y = "Popularity Score")

plot2 <- ggplot(ted_data, aes(video_age, popularity_score)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", formula = y ~ poly(x, 6), color = "red") +
  labs(title = "Curved Relationship between Video Age and Popularity",
       x = "Video Age (days)",
       y = "Popularity Score")

# Combine side-by-side
plot1
plot2
```

```
if (!requireNamespace("lmtest", quietly = TRUE)) {
  install.packages("lmtest")
,eval = FALSE}
library(lmtest)
resettest(full_model)
```

Choices of Degree:

```
final_model1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker +
                     poly(video_age, 2) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 2):languages + poly(video_age, 2):languages +
                     num_speaker:languages, data = ted_data)
#model_summary <- summary(final_model1)
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")

final_model1 <- lm(popularity_score ~ poly(duration, 3) + languages + num_speaker +
                     poly(video_age, 2) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 3):languages + poly(video_age, 2):languages +
                     num_speaker:languages, data = ted_data)
#model_summary <- summary(final_model1)
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")

final_model1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker +
                     poly(video_age, 3) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 2):languages + poly(video_age, 3):languages +
                     num_speaker:languages, data = ted_data)
#model_summary <- summary(final_model1)
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
```

```
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")

final_model1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker +
                     poly(video_age, 5) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 2):languages + poly(video_age, 5):languages +
                     num_speaker:languages, data = ted_data)
model_summary <- summary(final_model1)

#model_summary <- summary(final_model1)
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")

final_model <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker +
                     poly(video_age, 6) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 2):languages + poly(video_age, 6):languages +
                     num_speaker:languages, data = ted_data)
model_summary <- summary(final_model)

#model_summary <- summary(final_model1)
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")

final_model1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker +
                     poly(video_age, 7) + tag_count + Is_rating_t5_Inspiring +
                     tag_is_technology + tag_is_science + tag_is_global_issues +
                     poly(duration, 2):languages + poly(video_age, 7):languages +
                     num_speaker:languages, data = ted_data)
model_summary <- summary(final_model1)

#model_summary <- summary(final_model1)
```

```r
#cat(model_summary$r.squared, model_summary$adj.r.squared)
aic_val <- AIC(final_model1)
#cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model1)
#cat("BIC:", bic_val, "\n")
library(olsrr)
#cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
#cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model1) / (1 - hatvalues(final_model1)))^2)
#cat("PRESS:", press_val, "\n")
```

5 Model Selection

```r
step_model_aic <- step(final_model, direction = "both")
summary(step_model_aic)

#Main effect model
model_summary <- summary(full_model)
model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(full_model)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(full_model)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(full_model,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(full_model) / (1 - hatvalues(full_model)))^2)
cat("PRESS:", press_val, "\n")

#Full Model
model_summary <- summary(final_model)
model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(final_model)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(final_model)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(final_model,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(final_model) / (1 - hatvalues(final_model)))^2)
cat("PRESS:", press_val, "\n")
```

```
#Model after AIC
model_summary <- summary(step_model_aic)
model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(step_model_aic)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(step_model_aic)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(step_model_aic,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(step_model_aic) / (1 - hatvalues(step_model_aic)))^2)
cat("PRESS:", press_val, "\n")

#Step AIC stats:
step1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker + poly(video_age, 6) + tag_c
              tag_is_technology + tag_is_science + tag_is_global_issues + poly(duration, 2):languages +

#model_summary <- summary(step1)
#model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(step1)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(step1)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(step1,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(step1) / (1 - hatvalues(step1)))^2)
cat("PRESS:", press_val, "\n")

step1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker + poly(video_age, 6) + tag_c

#model_summary <- summary(step1)
#model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(step1)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(step1)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(step1,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(step1) / (1 - hatvalues(step1)))^2)
```

```
cat("PRESS:", press_val, "\n")

step1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker + poly(video_age, 6) + tag_c

#model_summary <- summary(step1)
#model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(step1)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(step1)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(step1,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(step1) / (1 - hatvalues(step1)))^2)
cat("PRESS:", press_val, "\n")

step1 <- lm(popularity_score ~ poly(duration, 2) + languages + num_speaker + poly(video_age, 6) + tag_c

#model_summary <- summary(step1)
#model_summary
R2 <- model_summary$r.squared
adj_R2 <- model_summary$adj.r.squared
cat("R-squared:", R2, "\n")
cat("Adjusted R-squared:", adj_R2, "\n")
aic_val <- AIC(step1)
cat("AIC:", aic_val, "\n")
bic_val <- BIC(step1)
cat("BIC:", bic_val, "\n")
library(olsrr)
cp_table <- ols_mallows_cp(step1,fullmodel = final_model)
cat("Cp", cp_table, "\n")
press_val <- sum((resid(step1) / (1 - hatvalues(step1)))^2)
cat("PRESS:", press_val, "\n")
```

5.2 Data Validation – Train/Test Split

```
# Splitting Data
set.seed(123)  # reproducibility
sample_size <- floor(0.8 * nrow(ted_data))
train_indices <- sample(seq_len(nrow(ted_data)), size = sample_size)

train_data <- ted_data[train_indices, ]
test_data <- ted_data[-train_indices, ]

# Model fitting (train)
model_train <- lm(popularity_score ~
                    poly(duration, 2) +
```

```r
                    languages +
                    num_speaker +
                    poly(video_age, 6) +
                    tag_count +
                    tag_is_science +
                    tag_is_global_issues +
                    poly(duration, 2):languages +
                    poly(video_age, 6):languages,
                  data = train_data)

# Prediction (test set)
test_preds <- predict(model_train, newdata = test_data)

# Actual values
actuals_test <- test_data$popularity_score
actuals_train <- train_data$popularity_score

# Compute validation statistics
RMSE <- sqrt(mean((actuals_test - test_preds)^2))
MAE <- mean(abs(actuals_test - test_preds))
R_squared <- cor(actuals_test, test_preds)^2
MSPR <- mean((actuals_test - test_preds)^2)    # Test error
MSE_F <- mean(residuals(model_train)^2)        # Training error (final model)

# Display results neatly
cat("Model Validation Statistics:\n")
cat(" - RMSE:", round(RMSE, 3), "\n")
cat(" - MAE:", round(MAE, 3), "\n")
cat(" - R-squared:", round(R_squared, 3), "\n")
cat(" - MSPR (Test MSE):", round(MSPR, 3), "\n")
cat(" - MSE_F (Train MSE):", round(MSE_F, 3), "\n")
```

6 Model Diagnostics

6.1 LINE Check

```r
par(mfrow = c(1,4))
plot(step_model_aic)
```

6.2 Outlier, Leverage, and Influence Diagnostics

```r
stud_resid <- rstudent(step_model_aic)
n <- nrow(model.frame(step_model_aic))
p.prime <- length(coef(step_model_aic))
alpha <- 0.05
t.crit <- qt(1 - (alpha/(2*n)), df = n - p.prime - 1)
potential_outliers <- which(abs(stud_resid) > t.crit)
cat("Outlier Detection Summary:\n",
```

```r
    "--------------------------\n",
    "Number of observations (n):", n, "\n",
    "Number of predictors (p'):", p.prime, "\n",
    "Bonferroni-adjusted critical value (t.crit):", round(t.crit, 3), "\n",
    "Number of potential outliers:", length(potential_outliers), "\n",
    "Potential outlier indices:", paste(potential_outliers, collapse = ", "), "\n")

-----------
leverages <- hatvalues(step_model_aic)
n <- nrow(model.frame(step_model_aic))
p.prime <- length(coef(step_model_aic))
leverage_threshold <- 2 * p.prime / n
high_leverage_indices <- which(leverages > leverage_threshold)
length(high_leverage_indices)  # See how many observations were flagged

#Cook's Distance
cooks_d <- cooks.distance(step_model_aic)
n <- nrow(model.frame(step_model_aic))
cook_threshold <- 4 / n
influential_cooks <- which(cooks_d > cook_threshold)
#influential_cooks # this is huge
length(influential_cooks)  # See how many observations were flagged
ols_plot_cooksd_chart(step_model_aic)
```