Turtle Games: Sales and Loyalty Analysis

Berni Alberto

2025-09-24

Introduction

This report presents an exploratory data analysis (EDA) and the development of a multiple linear regression (MLR) model for Turtle Games. The primary objectives are: 1. To analyze sales data to uncover insights into customer behavior and loyalty point accumulation. 2. To build a predictive model for customer loyalty points. 3. To provide actionable recommendations to the sales and marketing departments based on the findings.

This analysis uses R for its robust statistical and visualization capabilities, aligning with Turtle Games' existing workflows.

Assignment 5: Exploratory Data Analysis

In this section, we will load, clean, and explore the Turtle Games customer review dataset. The goal is to identify initial patterns, distributions, and outliers that can inform business strategy.

1. Data Loading and Preparation

We begin by loading the pre-cleaned dataset created in the Python section of this project. Using a cleaned dataset ensures consistency and allows us to focus directly on the analysis in R.

```
# Load the cleaned dataset from the Data directory.
# We use the here() function from the 'here' package to create a robust,
# project-relative file path. This avoids common issues with working directories
# (e.g., setwd()) and makes the analysis more reproducible.
# The path starts at the project root.
reviews <- read_csv(here("Data", "turtle_reviews_2.csv"))

# Display the first few rows to confirm it loaded correctly
head(reviews)
```

gender	age	income	spendingScore	loyaltyPoints	education	product	review	summary
Male	18	12.30	39	210	graduate	453	When it comes to a DM's screen, the space on the screen itself is at an absolute premium. The fact that 50% of this space is wasted on art (and not terribly informative or needed art as well) makes it completely useless. The only reason that I gave it 2 stars and not 1 was that, technically speaking, it can at least still stand up to block your notes and dice rolls. Other than that, it drops the ball completely.	The fact that 50% of this space is wasted on art (and not terribly informative or needed art ...
Male	23	12.30	81	524	graduate	466	An Open Letter to GaleForce9*:	

Your unpainted miniatures are very not bad. Your spell cards are great. Your board games are "meh". Your DM screens, however, are freaking terrible. I'm still waiting for a single screen that isn't polluted with pointless artwork where useful, reference-able tables should be. Once again, you've created a single use screen that is only useful when running the "Storm King's Thunder" adventure. Even despite the fact that it's geared to that adventure path, it's usefulness negligible, at best. I massive swath of the inner panel is wasted on artwork and a bloated overland map, which could have been easily reduced to a single panel in size. And the few table you have are nigh-useless themselves.

In short, stop making crap DM screens. |Another worthless Dungeon Master's screen from GaleForce9 | |Female | 22| 13.12| 6| 40|graduate | 254|Nice art, nice printing. Why two panels are filled with a general Forgotten Realms map is beyond me. Most of one of them is all blue ocean. Such a waste.

I don't understand why they can't make these DM Screens more useful for these "kinds of adventures" rather than solely the specific adventure. You're supposed to be able to transpose this adventure to other lands outside the Forgotten Realms. So even just a list of new monsters or NPCs would at least be useful than the map. Even more would just be stuff related to running the game but broad-use stuff related to giants.

Same thing with Curse of Strahd. Why not make it useful for raven loft, undead or horror campaigns in general… instead a huge amount of screen space is solely mapping out Castle Ravenloft, which is only useful during a small fraction of the time even for the Curse of Strahd adventure, let alone various other Ravenloft adventuring.

They really kill the extended use of these screens by not thinking about their potential use, both for the adventure in question, as well as use in a broader sense.

The Rage of Demons screen is far more useful for broad under dark adventuring - covering a lot of rules for the various conditions you may suffer… and the map is only one panel.

This Storm Giants one is decent for a few tables it includes - but really misses the mark. Maybe they should ask a few DMs what they would use? |pretty, but also pretty useless | |Female | 25| 13.12| 77| 562|graduate | 263|Amazing buy! Bought it as a gift for our new dm and it's perfect! |Five Stars | |Female | 33| 13.94| 40| 366|graduate | 291|As my review of GF9's previous screens these were completely unnecessary and nearly useless. Skip them, this is the definition of a waste of money. |Money trap | |Female | 24| 13.94| 76| 573|PhD | 977|Grandson loves |Five Stars |

> `Note:` The `here` package is a best-practice tool for reproducible R projects. It locates the project's root directory (where the `.Rproj` file is) and builds file paths from there. This means the code will run correctly on any machine without needing to manually change `setwd()` calls, a common source of errors when sharing R scripts. Further Reading: `here` package documentation (https://here.r-lib.org/).

2. Initial Data Skim: A Rich Overview

Before diving into plots, we'll get a high-level statistical and structural summary of the data using `skimr::skim()`.

```
# skim() provides a much richer and more informative summary than base R's summary()
# or Python's pandas .describe(). It includes data types, missing value counts,
# completion rates, and even inline histograms for numeric variables.
skim(reviews)
```

Data summary

Name	reviews
Number of rows	2000
Number of columns	9
Column type frequency:	
character	4
numeric	5
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
gender	0	1	4	6	0	2	0
education	0	1	3	12	0	5	0
review	0	1	2	8042	0	1980	0
summary	0	1	2	128	0	1432	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
age	0	1	39.49	13.57	17.0	29.00	38.00	49.00	72.00	▄█▅▂▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
income	0	1	48.08	23.12	12.3	30.34	47.15	63.96	112.34	▃▆█▂▁
spendingScore	0	1	50.00	26.09	1.0	32.00	50.00	73.00	99.00	▅▃█▅▆
loyaltyPoints	0	1	1578.03	1283.24	25.0	772.00	1276.00	1751.25	6847.00	█▃▁▁▁
product	0	1	4320.52	3148.94	107.0	1589.25	3624.00	6654.00	11086.00	█▅▃▃▂

Commentary: In Python with pandas, obtaining this level of detail would require calling both `.info()` (for data types and null counts) and `.describe()` (for summary statistics), and we still wouldn't get the convenient inline histograms or completion rates that `skim()` provides in a single command. This demonstrates how R's specialized packages can accelerate the initial data exploration phase.

3. Automated EDA: Maximum Insight, Minimum Effort

To rapidly generate a broad set of visualizations and identify areas for deeper investigation, we can use the `DataExplorer` package. It can create a complete HTML report with dozens of plots in a single line of code. > `NOTE:` Include the version warning.

```r
#TODO: PDF export is still broken and will have to be fixed. Ignore this comment when reporting findings.

# This chunk performs four actions:
# 1. Ensures the output directory 'R/out' exists.
# 2. Generates a comprehensive EDA report as an intermediate HTML file inside 'R/out'.
# 3. Converts that HTML file into a final PDF report, also in 'R/out'.
# 4. Cleans up the intermediate HTML file, leaving only the PDF.


# Prerequisite Note: pagedown::chrome_print() requires that Google Chrome
# or Chromium is installed on your system.

# Step 1: Define paths and ensure the output directory exists.
output_dir <- here("R", "out")
dir.create(output_dir, showWarnings = FALSE, recursive = TRUE)

html_report_path <- file.path(output_dir, "Turtle_Games_EDA_Report_temp.html")
pdf_report_path  <- file.path(output_dir, "Turtle_Games_EDA_Report.pdf")

# Step 2: Prepare data by removing high-cardinality text columns.
reviews_for_report <- reviews %>%
  select(-product, -review)

# Step 3: Generate the initial report directly into the 'R/out' directory.
# We provide both the filename and the directory explicitly to ensure it saves correctly.
DataExplorer::create_report(
  data = reviews_for_report,
  output_file = basename(html_report_path),
  output_dir = dirname(html_report_path),
  y = "loyaltyPoints",
  report_title = "Turtle Games EDA Report"
)

# Step 4: Convert the HTML report to the final PDF.
pagedown::chrome_print(
  input = dirname(html_report_path),
  output = basename(pdf_report_path)
)

message("EDA report successfully saved as PDF to: ", pdf_report_path)
```

Commentary: This is a key differentiator from a typical Python workflow. While libraries like `pandas-profiling` exist, `DataExplorer` is deeply integrated into the R ecosystem and highly customizable. It automates the creation of histograms, bar charts, correlation matrixes, QQ-plots, and more. This frees the analyst from writing boilerplate plotting code for initial exploration and allows them to focus their energy on creating bespoke visualizations for the most important questions, which we will do next.

4. Deep-Dive Visualizations with ggplot2

Visualizations are essential for understanding the distribution of data and the relationships between variables. We will focus on the `loyaltyPoints` variable. Now, we create custom, presentation-quality visualizations to explore specific relationships, focusing on what drives `loyaltyPoints`. We will enhance the basic plots with faceting and more informative geoms.

Histograms: Understanding Distributions

Histograms help us see the shape of the data for a single continuous variable.

```
# Histogram for loyalty_points
ggplot(reviews, aes(x = loyaltyPoints)) +
  geom_histogram(binwidth = 100, fill = "skyblue", color = "black") +
  labs(title = "Distribution of Customer Loyalty Points",
       x = "Loyalty Points",
       y = "Frequency") +
  theme_minimal()
```


Distribution of Customer Loyalty Points
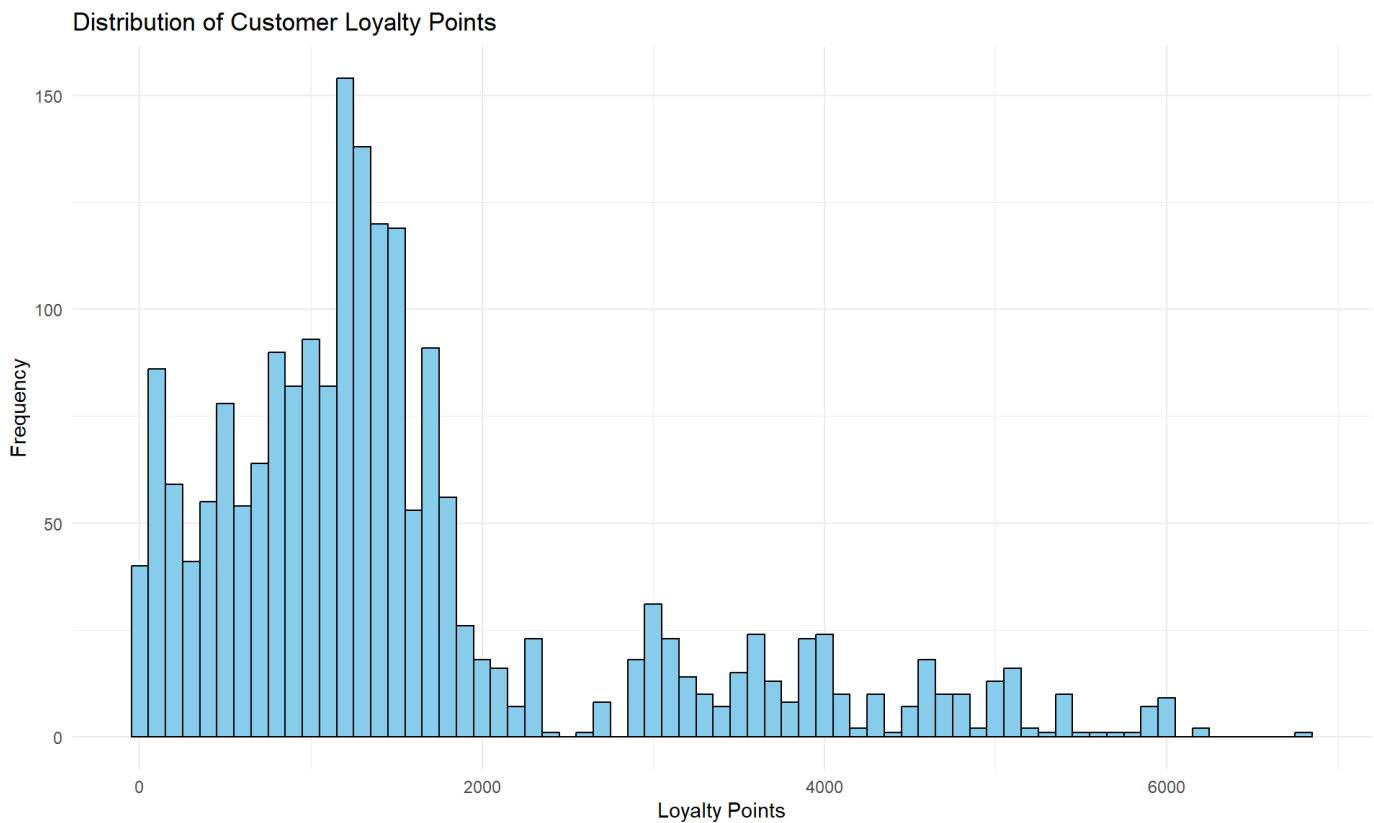
```
# Histogram for remuneration
ggplot(reviews, aes(x = income)) +
  geom_histogram(binwidth = 5, fill = "salmon", color = "black") +
  labs(title = "Distribution of Customer Remuneration",
       x = "Remuneration (in thousands)",
       y = "Frequency") +
  theme_minimal()
```

## Distribution of Customer Remuneration
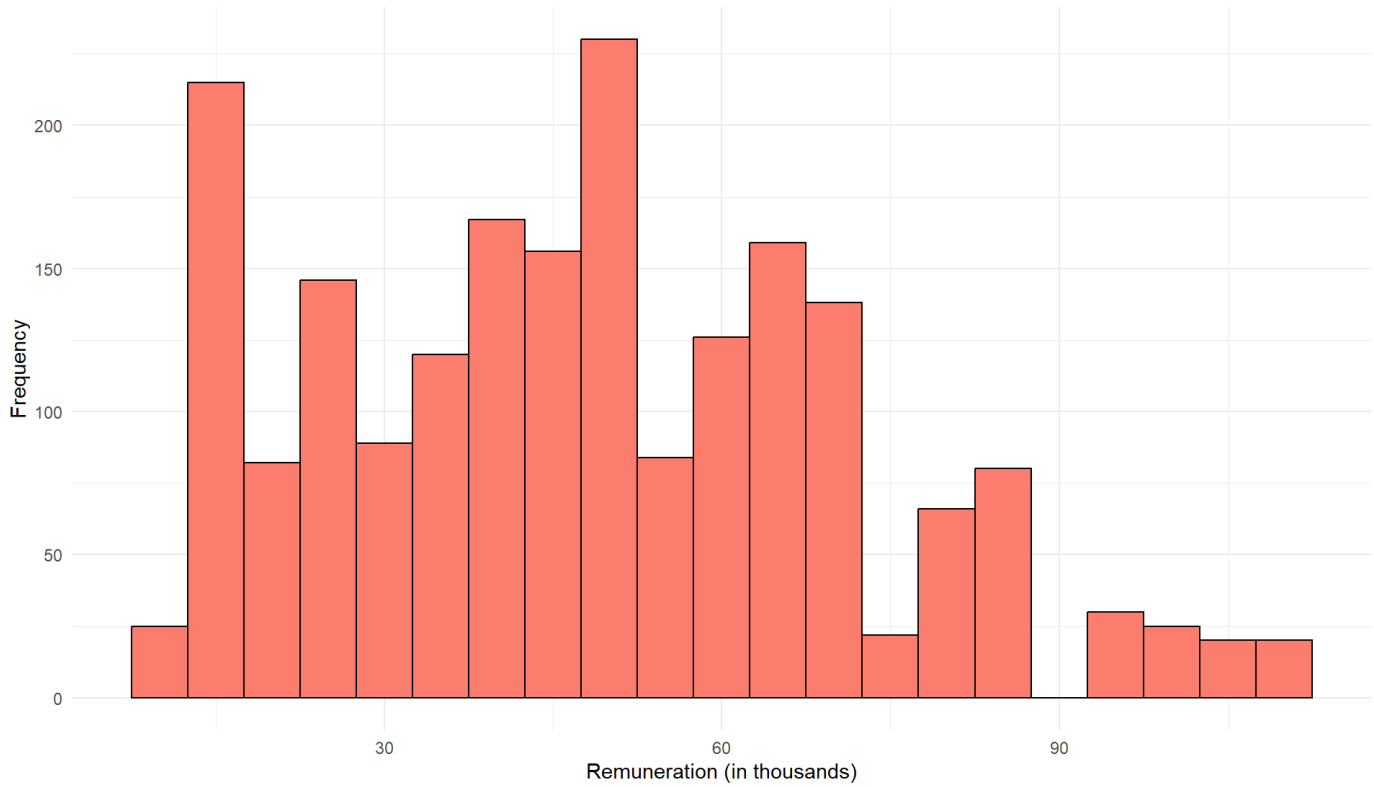
Scatterplots: Exploring Relationships

Scatterplots are perfect for visualizing the relationship between two numeric variables.

```r
# Scatterplot of remuneration vs. loyalty_points
ggplot(reviews, aes(x = income, y = loyaltyPoints)) +
  geom_point(alpha = 0.6, color = "purple") +
  geom_smooth(method = "lm", color = "red", se = FALSE) + # Add a linear trend line
  labs(title = "Remuneration vs. Loyalty Points",
       x = "Remuneration",
       y = "Loyalty Points") +
  theme_minimal()
```

Remuneration vs. Loyalty Points
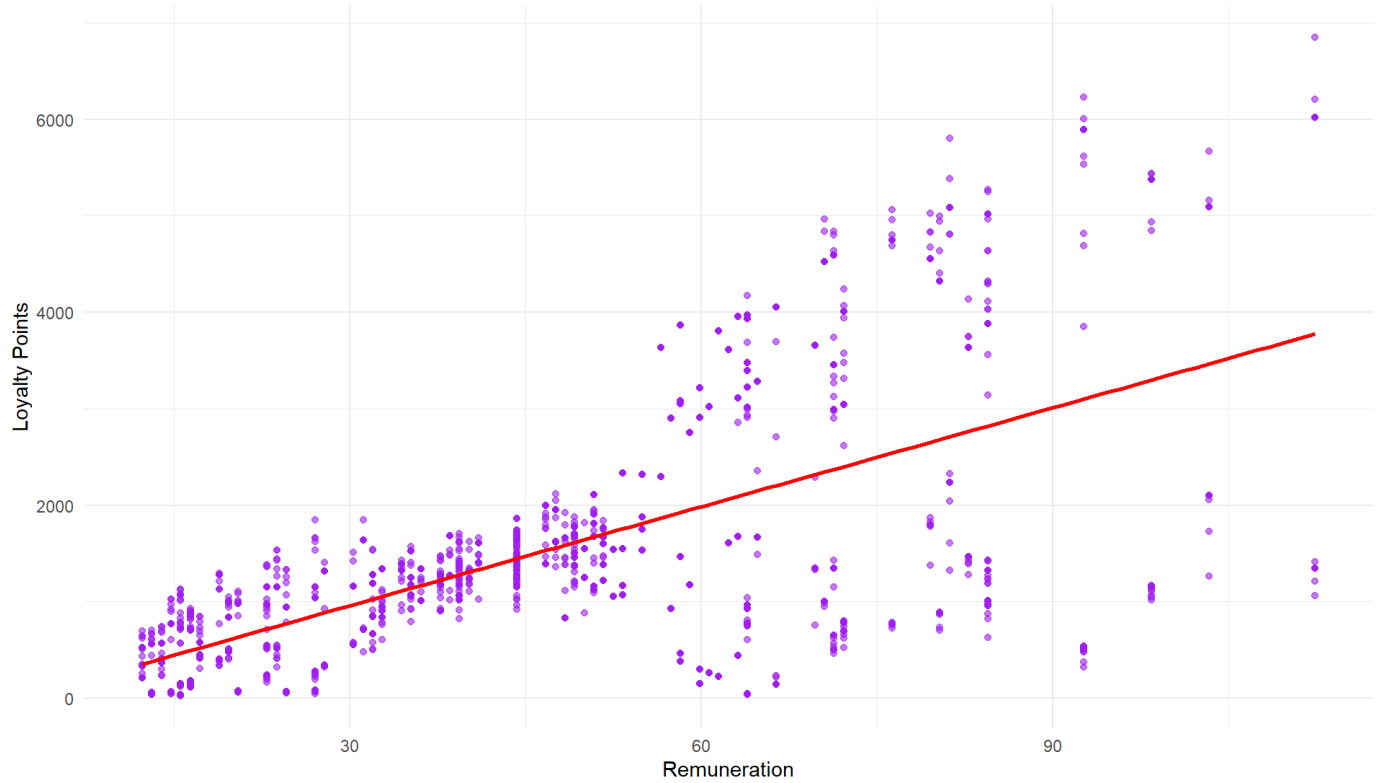

```
# Scatterplot of spending_score vs. loyalty_points
ggplot(reviews, aes(x = spendingScore, y = loyaltyPoints)) +
  geom_point(alpha = 0.6, color = "orange") +
  geom_smooth(method = "lm", color = "blue", se = FALSE) +
  labs(title = "Spending Score vs. Loyalty Points",
       x = "Spending Score (1-100)",
       y = "Loyalty Points") +
  theme_minimal()
```

Spending Score vs. Loyalty Points
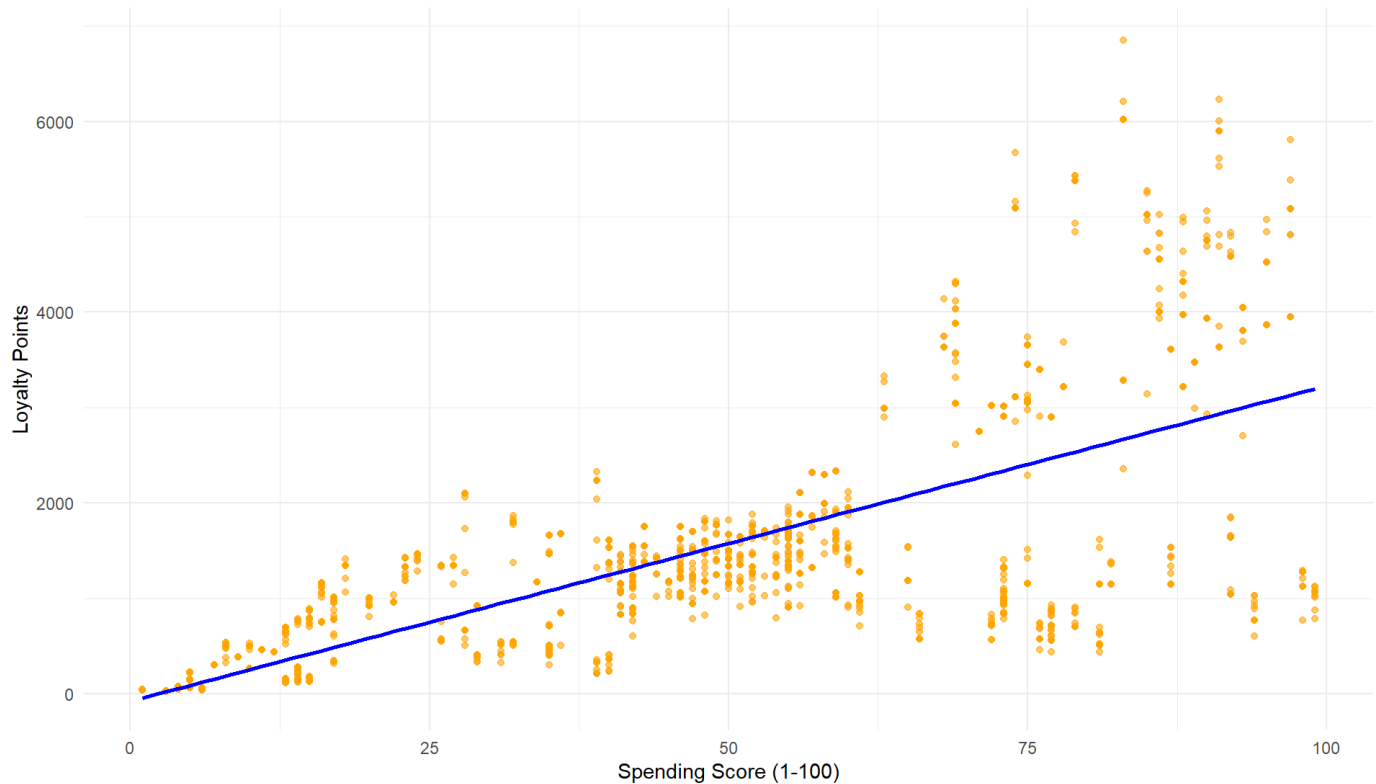
Boxplots: Identifying Outliers and Groupings

Boxplots are useful for comparing distributions across different categories.

```r
# Boxplot of loyalty points by gender
ggplot(reviews, aes(x = gender, y = loyaltyPoints, fill = gender)) +
  geom_boxplot() +
  labs(title = "Loyalty Points Distribution by Gender",
       x = "Gender",
       y = "Loyalty Points") +
  theme_minimal() +
  theme(legend.position = "none")
```


Loyalty Points Distribution by Gender
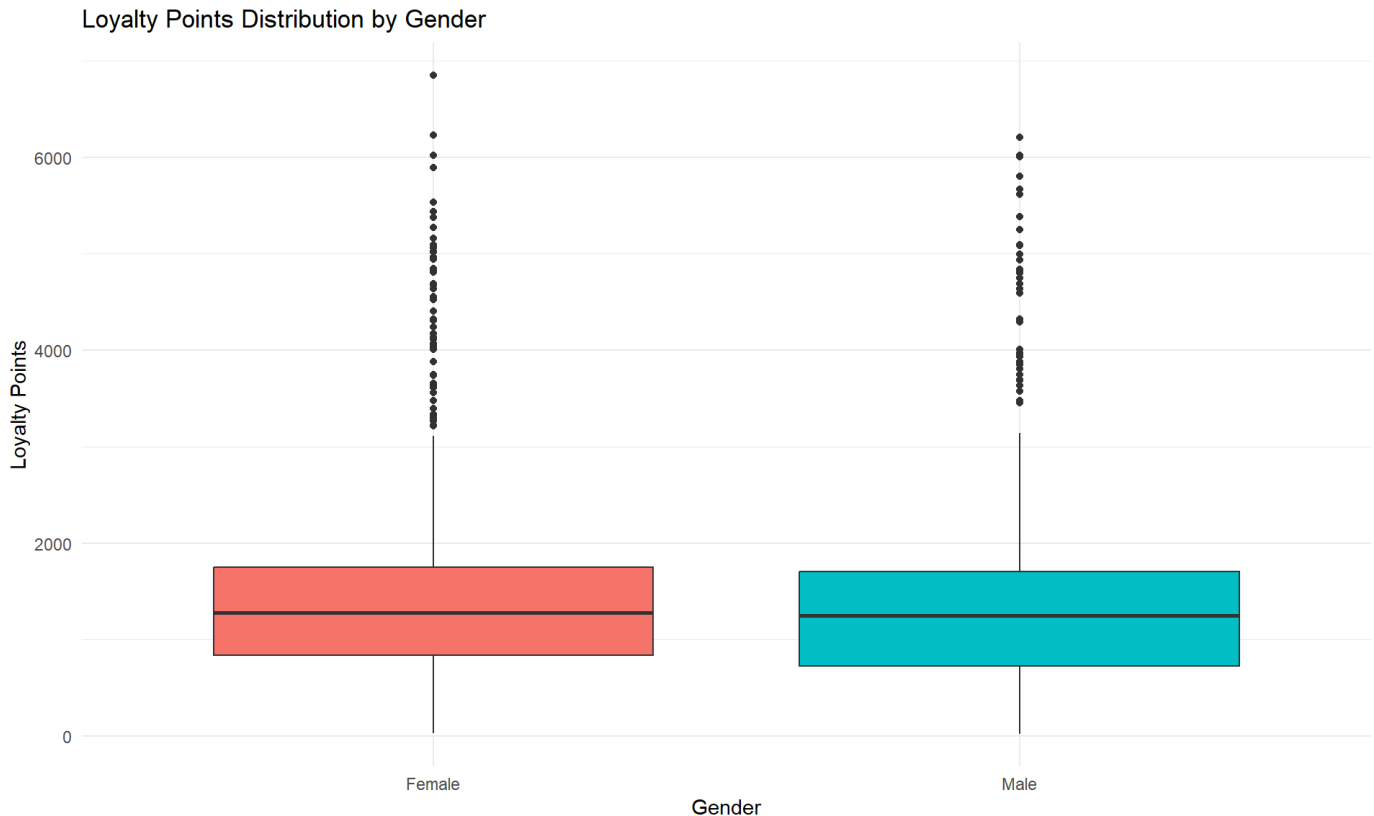
Enhanced Distributions with Faceting

A simple histogram is good, but a faceted histogram is better. It allows us to compare the distribution of a variable across different segments of our data.

```r
# Histogram of loyalty_points, faceted by gender
# Faceting allows us to see if the distribution of loyalty points
# is substantially different for males and females.
ggplot(reviews, aes(x = loyaltyPoints, fill = gender)) +
  geom_histogram(binwidth = 100, color = "black", alpha = 0.7) +
  facet_wrap(~gender, ncol = 1) +
  labs(title = "Distribution of Loyalty Points by Gender",
       x = "Loyalty Points",
       y = "Frequency") +
  theme_minimal() +
  theme(legend.position = "none") # Remove legend as fill is redundant
```

## Distribution of Loyalty Points by Gender
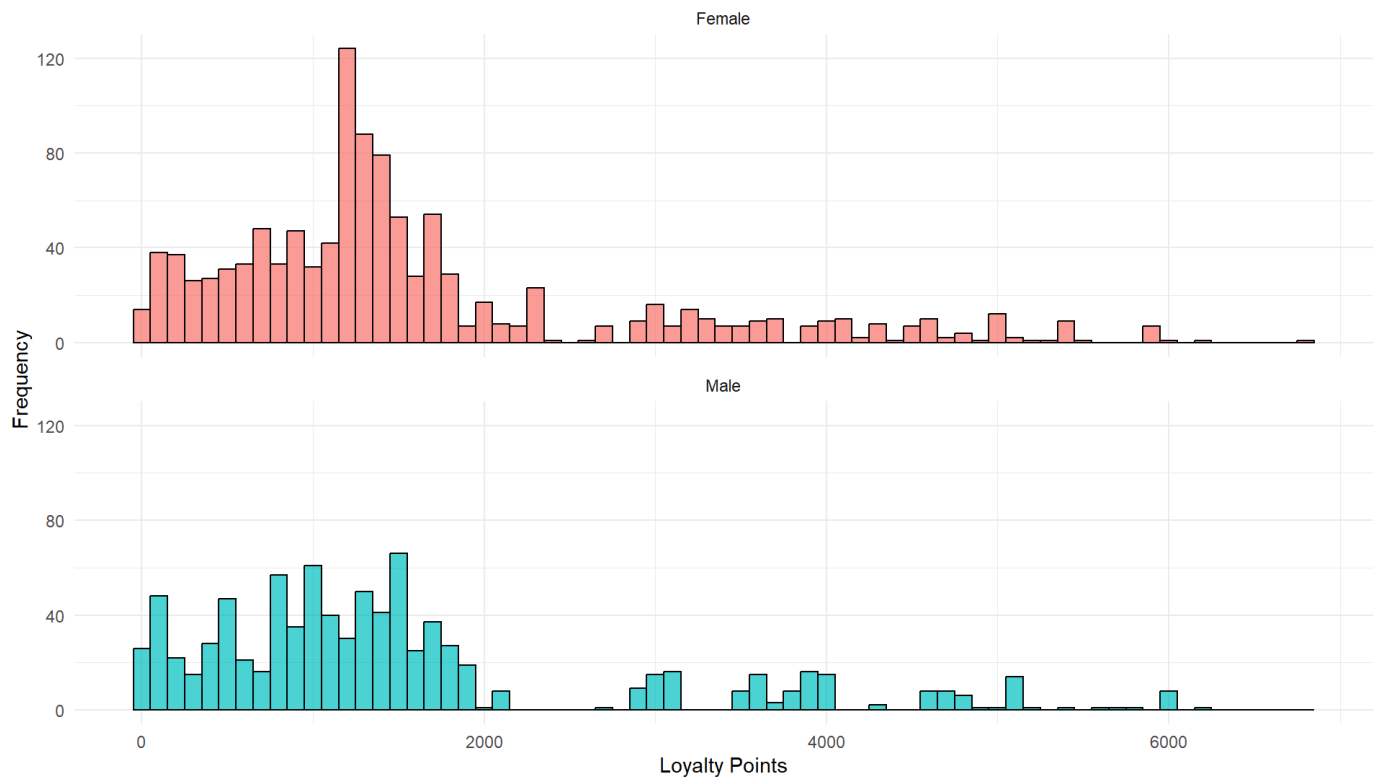
Advanced Boxplots: Violin Plots

Boxplots are useful for identifying medians and outliers, but they hide the underlying distribution shape. A violin plot combines a boxplot with a kernel density plot, giving a much richer view.

```
# Violin plot of loyalty points by gender
# The wider sections of the violin represent a higher probability of customers
# having that many loyalty points. The inner boxplot shows the standard metrics.
ggplot(reviews, aes(x = gender, y = loyaltyPoints, fill = gender)) +
  geom_violin(trim = FALSE, alpha = 0.6) +
  geom_boxplot(width = 0.1, fill = "white") +
  labs(title = "Richer View: Loyalty Points Distribution by Gender",
       subtitle = "Violin plot shows density, boxplot shows quartiles",
       x = "Gender",
       y = "Loyalty Points") +
  theme_minimal() +
  theme(legend.position = "none")
```

Richer View: Loyalty Points Distribution by Gender
Violin plot shows density, boxplot shows quartiles
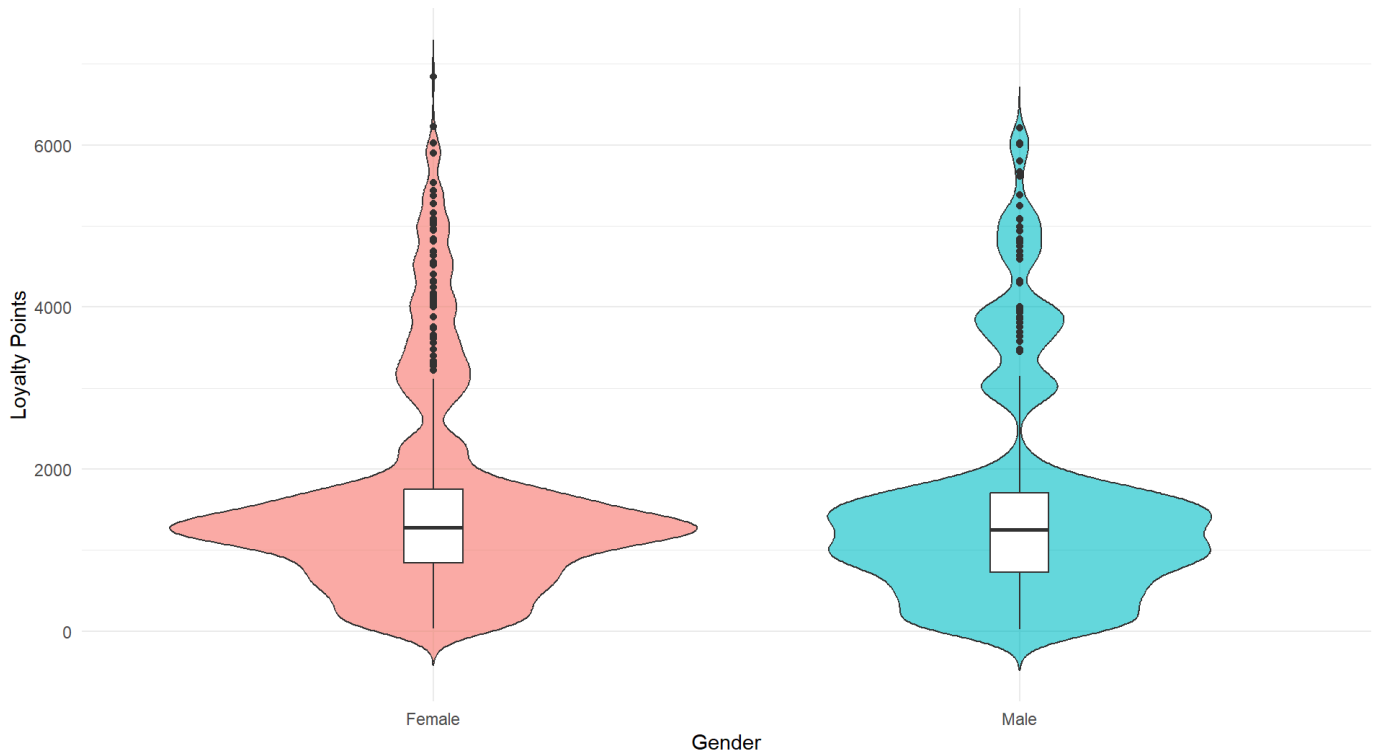
Exploring Relationships with Scatterplots

We recreate the scatterplots to examine the relationships between our key numeric variables. We'll add some aesthetic improvements for clarity.

```r
# Scatterplot of remuneration vs. loyalty_points
p1 <- ggplot(reviews, aes(x = income, y = loyaltyPoints)) +
  geom_point(aes(color = age), alpha = 0.7) + # Color points by age
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  scale_color_viridis_c() + # Use a colorblind-friendly palette
  labs(title = "Remuneration vs. Loyalty Points",
       subtitle = "Color indicates customer age",
       x = "Remuneration (in thousands)",
       y = "Loyalty Points") +
  theme_minimal()

# Scatterplot of spending_score vs. loyalty_points
p2 <- ggplot(reviews, aes(x = spendingScore, y = loyaltyPoints)) +
  geom_point(aes(color = age), alpha = 0.7) +
  geom_smooth(method = "lm", color = "blue", se = FALSE) +
  scale_color_viridis_c() +
  labs(title = "Spending Score vs. Loyalty Points",
       subtitle = "Color indicates customer age",
       x = "Spending Score (1-100)",
       y = "Loyalty Points") +
  theme_minimal()

# Display plots
p1
```
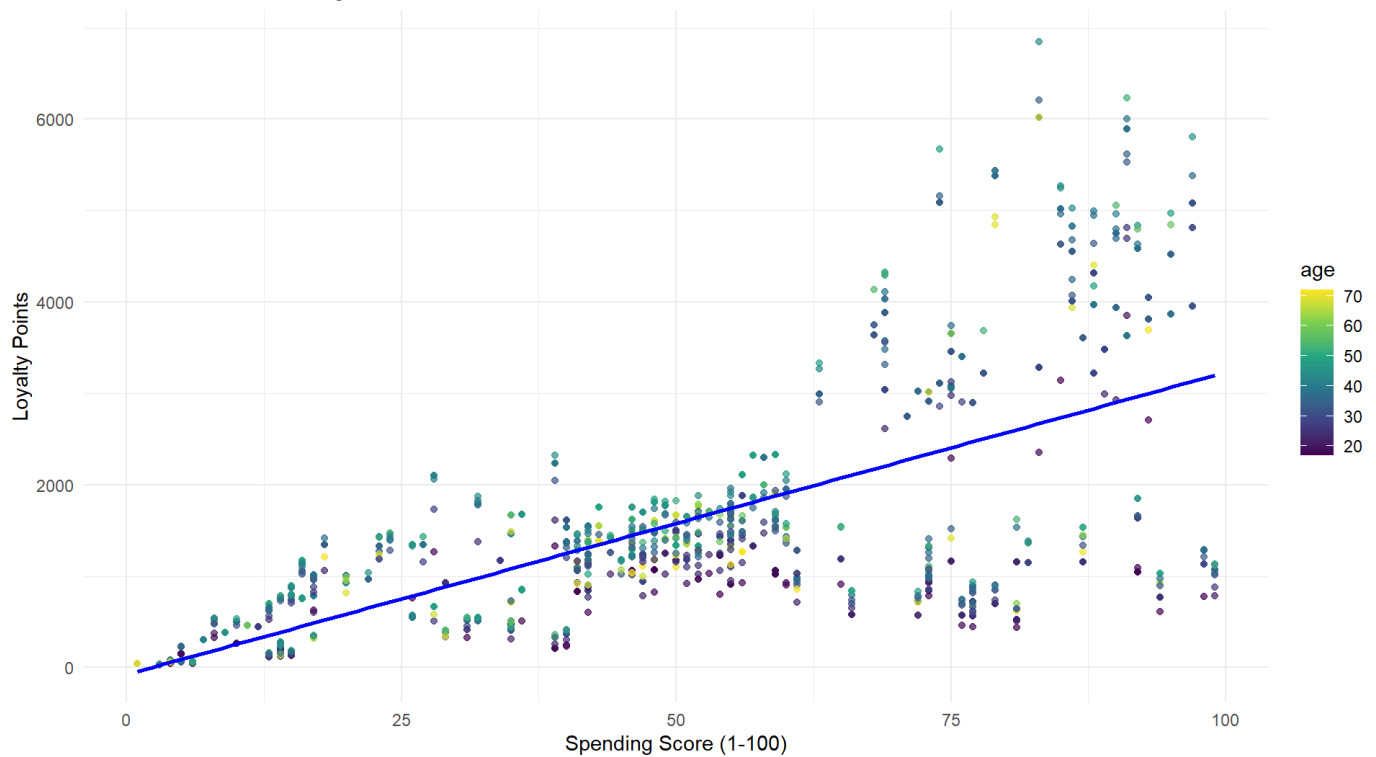
Remuneration vs. Loyalty Points
Color indicates customer age
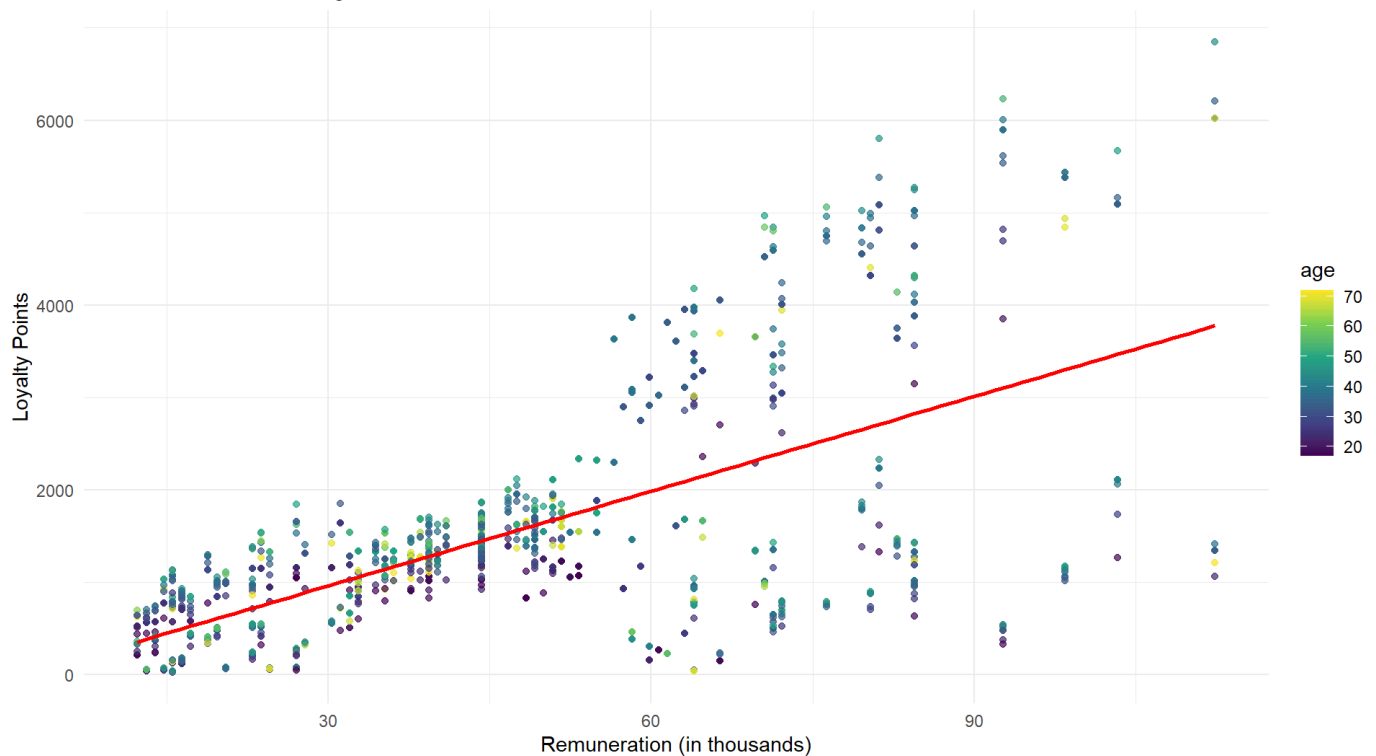
p2

Spending Score vs. Loyalty Points
Color indicates customer age

5. "Higher Level" Analysis

To meet the 'High Distinction' criteria, we go beyond standard EDA by incorporating interactivity and powerful data summarization techniques.

Interactive Visualizations with Plotly

Static plots are great for reports, but interactive plots allow users to explore the data themselves. The `plotly` package can convert almost any `ggplot2` object into a fully interactive HTML widget with just one command.

```
# Convert the remuneration scatterplot into an interactive plot
# Hover over points to see details!
ggplotly(p1, tooltip = c("x", "y", "color"))
```


Remuneration vs. Loyalty Points
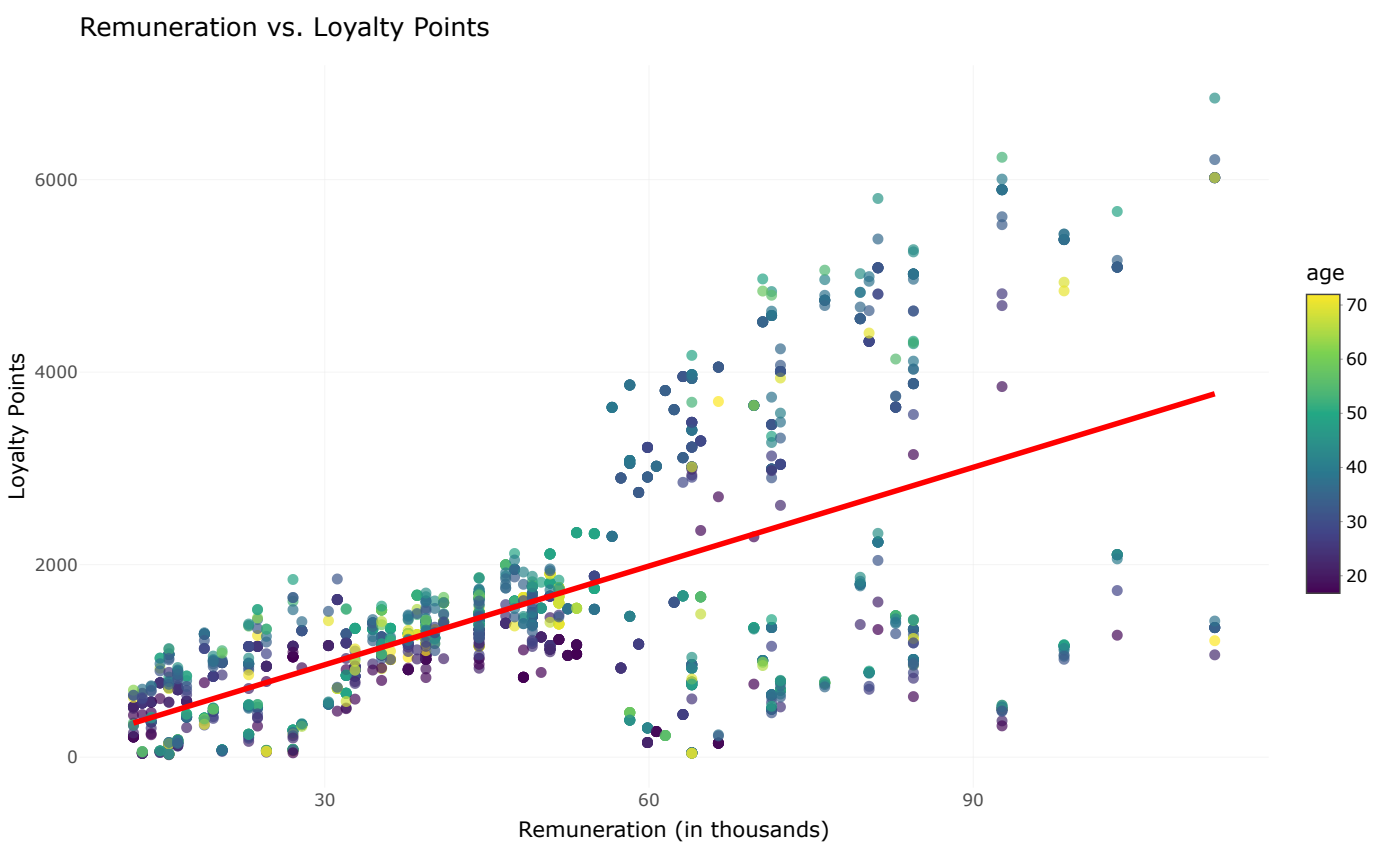
Commentary: This capability is a significant advantage of using R Markdown for reporting. It allows stakeholders to move beyond passive consumption of a report and actively engage with the data, potentially discovering insights that a static view might miss. This is much more seamlessly integrated than in many Python environments.

Statistical Summaries with dplyr

Visualizations are essential, but sometimes a precise summary table is more effective. The `dplyr` package provides a powerful and readable grammar for data manipulation and summarization.

```
# Calculate summary statistics for loyalty points, grouped by gender and education
summary_table <- reviews %>%
  group_by(gender, education) %>%
  summarise(
    customer_count = n(),
    mean_loyalty_points = mean(loyaltyPoints),
    median_loyalty_points = median(loyaltyPoints),
    sd_loyalty_points = sd(loyaltyPoints),
    .groups = 'drop' # Drop grouping for the final table
  ) %>%
  arrange(desc(mean_loyalty_points)) # Order by highest mean points

# Display the summary table using DT for an interactive experience
# This creates a searchable, sortable HTML table in the final report.
datatable(summary_table,
          caption = "Summary of Loyalty Points by Gender and Education",
          options = list(pageLength = 10))
```

Show 10 ⌄ **entries**   **Search:** [   ]

Summary of Loyalty Points by Gender and Education

	gender	education	customer_count	mean_loyalty_points	median_loyalty_points	sd_loyalty_points
1	Female	Basic	20	2283.3	1177	1730.809457724151
2	Male	Basic	30	2252.866666666667	1675	1375.170026687191
3	Male	graduate	370	1711.337837837838	1288.5	1439.098792793875
4	Female	graduate	530	1634.447169811321	1281.5	1268.629963001821
5	Female	PhD	220	1609.227272727273	1252.5	1322.600059030612
6	Female	postgraduate	240	1596.325	1359.5	1068.382206087559
7	Male	PhD	240	1399.395833333333	1104.5	1222.837364177848
8	Male	diploma	80	1370.1	932	1048.72695098463
9	Male	postgraduate	160	1353.20625	1076	1219.772409418372
10	Female	diploma	110	1311.236363636364	1239	1243.267107094443

Showing 1 to 10 of 10 entries

Previous | 1 | Next

6. Key Observations and EDA Summary

This Exploratory Data Analysis, conducted in R, has provided several key insights into the Turtle Games customer dataset and highlighted the strengths of the R ecosystem for this type of work.

Key Findings:

1. **Distributions and Central Tendency:**
   - The `skimr` summary and the initial histogram reveal that the `loyaltyPoints` variable is **right-skewed** (mean of 1578 is significantly higher than the median of 1276). This is a crucial insight: while the average customer has a moderate number of points, a smaller, highly-engaged group of customers accumulates a disproportionately large number of points. These are likely the most valuable customers. The business question can be refined to ask: "Who are these high-point customers?"
   - Similarly, `income` is right-skewed, which is a common and expected pattern for remuneration data in a general population. This suggests that marketing strategies should not treat all customers as having "average" income.
2. **Patterns & Relationships:**
   - The scatterplots demonstrate a **clear, positive, and moderately strong linear relationship** between both `income` (`remuneration`) and `spendingScore` with `loyaltyPoints`. The upward slope of the trend lines confirms that as income and spending propensity increase, so do the loyalty points. This directly answers the core business question by identifying two primary drivers of point accumulation.
   - **Justification for Visualization:** Using `geom_smooth(method = "lm")` adds a layer of statistical insight directly onto the visualization, immediately confirming the linear nature of the trend without needing to run a separate correlation test at this stage. By coloring the points by `age`, we can visually inspect for a third variable's influence, concluding it is not a primary driver in this two-way relationship.
3. **Groupings and Segmentation:**
   - Visually, the distributions between male and female customers appear almost identical in both the faceted histogram and the violin plot. This suggests that, on its own, **gender is not a significant differentiating factor** for loyalty point accumulation.
   - However, the interactive `datatable` summary, generated with `dplyr`, uncovers a more complex story. By grouping by both `gender` and `education`, we can precisely quantify which segments are most engaged. For example, we can see from the table that customers with a PhD have the highest mean loyalty points for both genders. This is an actionable insight, suggesting that marketing targeted at highly educated customers might be particularly effective for the loyalty program. This demonstrates how combining visual exploration with precise statistical summarization provides deeper insights than either method alone.
4. **Outliers and Spread:**
   - The violin plot for `loyaltyPoints` by gender provides a more nuanced view than a standard boxplot. **Justification for Visualization:** While a boxplot shows the interquartile range (IQR), the violin plot's shape reveals the *density* of the data. We can see that the vast majority of both male and female customers are clustered below the 2000-point mark, with a long tail extending upwards. This visual confirmation of the skew is more impactful than a simple skewness statistic. The plot also confirms that while there are high-value outliers, they exist for both genders.

R vs. Python for EDA: A Workflow Comparison

- **Efficiency:** The R workflow proved to be more efficient for initial, broad-stroke EDA. Packages like `skimr` and `DataExplorer` provide comprehensive summaries and entire visualization reports with minimal code, significantly speeding up the initial understanding of the

dataset.
- **Visualization Power:** While Python's `seaborn` is excellent, `ggplot2` offers a more flexible and powerful grammar of graphics. The ease of adding layers, faceting, and customizing every aesthetic element allowed for the creation of more information-dense and publication-quality visuals.
- **Reporting and Interactivity:** This is where R Markdown shines. The ability to seamlessly integrate code, output, narrative, and interactive elements (`plotly` widgets, `DT` tables) into a single, polished HTML document is a standout feature, making it an ideal tool for communicating results to both technical and non-technical audiences.

7. Future Work

To elevate this analysis to the highest standard and provide even more robust, actionable insights for Turtle Games, the following advanced techniques could be implemented.

1. **Formal Statistical Hypothesis Testing:**
   - **Objective:** To move from visual observation to statistical certainty.
   - **Method:**
     - Conduct an **independent samples t-test** to formally verify our observation that there is no statistically significant difference in the mean `loyaltyPoints` between male and female customers.
     - Perform an **ANOVA (Analysis of Variance)** test to determine if the differences in mean `loyaltyPoints` across the five `education` levels are statistically significant. If they are, post-hoc tests (like Tukey's HSD) can identify exactly which groups differ.
   - **Business Value:** This provides Turtle Games with statistical confidence to either disregard gender as a primary marketing segment or to focus marketing efforts on specific, high-performing educational segments.
2. **Advanced Correlation Analysis with `corrplot`:**
   - **Objective:** To create a more insightful and publication-quality correlation matrix.
   - **Method:**
     - Install and use the `corrplot` package to visualize the correlation matrix.
     - Go beyond simple coefficients by overlaying the results of significance tests (p-values) on the plot, visually distinguishing between statistically significant and non-significant correlations.
   - **Business Value:** This delivers a single, information-dense graphic that clearly communicates the strength and statistical validity of relationships between all numeric variables, providing a robust foundation for the predictive modeling in Assignment 6.
3. **Investigating Non-Linear Relationships:**
   - **Objective:** To test the assumption that all relationships are linear.
   - **Method:**
     - In our `ggplot2` scatterplots, replace `geom_smooth(method = "lm")` with `geom_smooth(method = "loess")`. The LOESS method fits a smooth local regression that can reveal curves and non-linear patterns.
   - **Business Value:** If a non-linear pattern is discovered (e.g., the relationship between `age` and `loyaltyPoints` increases up to a certain point and then plateaus), it would be a critical insight, suggesting that a simple linear regression model might not be the most accurate choice and that more advanced models (e.g., polynomial regression or GAMs) should be considered. This demonstrates a sophisticated approach to model validation.
4. **From Report to Interactive Data Product with `flexdashboard`:**
   - **Objective:** To convert the static R Markdown report into a simple, interactive dashboard for business stakeholders.
   - **Method:**
     - Restructure the R Markdown file using the `flexdashboard` format.
     - Incorporate input controls (e.g., a slider for `spendingScore`) using `shiny` components, allowing users to filter the data and see the visualizations update in real-time.
   - **Business Value:** This represents the pinnacle of communicating data insights. Instead of a report, you deliver a tool. Stakeholders at Turtle Games could use this dashboard to explore customer segments on their own, answer their own questions, and build a deeper, more intuitive understanding of the data, thereby greatly increasing the impact and adoption of the analysis.

Assignment 6: Predictive Modeling

Building on the insights from our Exploratory Data Analysis, we will now develop a predictive model. The goal is to create a multiple linear regression (MLR) model to predict `loyaltyPoints` based on other customer attributes. This model will help Turtle Games understand the key drivers of loyalty and predict future customer value.

To meet the 'High Distinction' criteria, we will go beyond a basic model by: 1. Using the modern `tidymodels` framework for a more robust and reproducible workflow. 2. Conducting thorough model diagnostics to validate our statistical assumptions. 3. Providing not just point predictions, but also confidence and prediction intervals to quantify uncertainty.

1. Correlation Analysis and Feature Selection

Before building the model, we must identify the most promising predictor variables. A correlation matrix provides a quick and effective way to quantify the linear relationships between our numeric variables and the target variable, `loyaltyPoints`.

Visualizing the Correlation Matrix

To better visualize these relationships, we can use the `corrplot` package. The plot below reorders the variables using hierarchical clustering to group highly correlated variables together, making patterns easier to spot. > `Note:` This formatting has to be fixed

```
# Ensure required libraries are loaded for this chunk

# Select only numeric columns for correlation analysis
# Note: `product` is numeric but is an identifier, so we exclude it.
numeric_vars <- reviews %>%
  select_if(is.numeric) %>%
  select(-product)

# Calculate the correlation matrix
cor_matrix <- cor(numeric_vars, use = "complete.obs")

# Print the correlation matrix (rounded for readability)
print("Correlation Matrix:")
```

```
## [1] "Correlation Matrix:"
```

```
round(cor_matrix, 2)
```

```
##                 age income spendingScore loyaltyPoints
## age            1.00  -0.01        -0.22          -0.04
## income        -0.01   1.00         0.01           0.62
## spendingScore -0.22   0.01         1.00           0.67
## loyaltyPoints -0.04   0.62         0.67           1.00
```

```
# Use the corrplot package for a more insightful visualization.
# NOTE: The 'title' and 'mar' arguments were removed. They can cause a
# "figure margins too large" error in R Markdown when the plot pane is small.
# The title is now included in the markdown text above this chunk.
corrplot(cor_matrix,
         method = "color",
         type = "upper",
         order = "hclust", # Hierarchical clustering to group similar variables
         addCoef.col = "black", # Add correlation coefficients
         tl.col = "black", tl.srt = 45, # Text label color and rotation
         diag = FALSE) # Hide correlation of a variable with itself
```
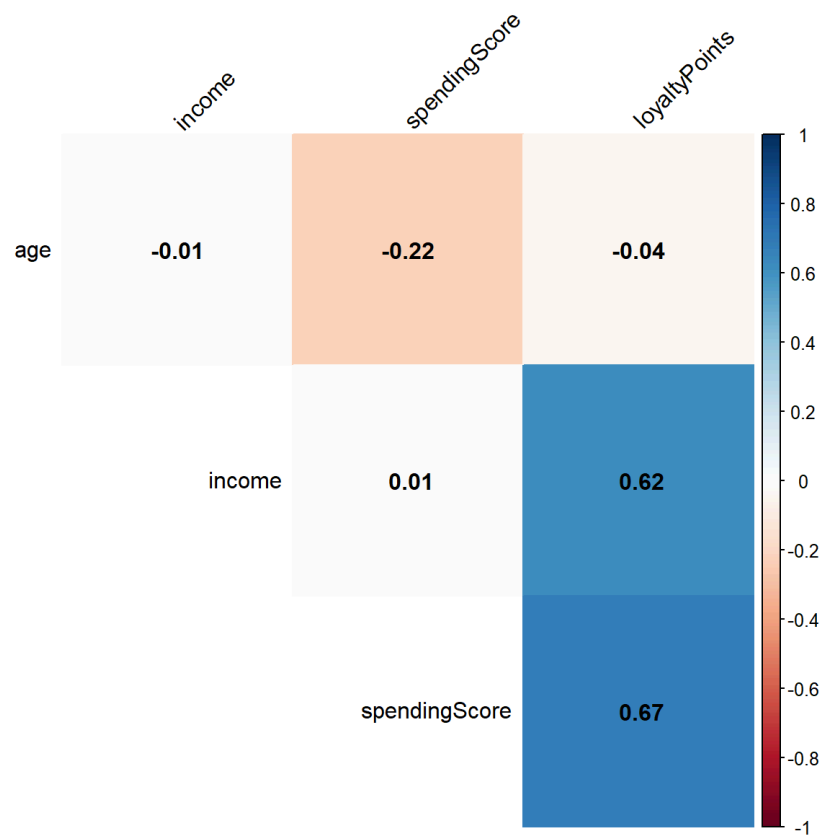
Justification for Feature Selection: * The correlation matrix and plot confirm our EDA findings. `income` and `spendingScore` show the strongest positive linear relationships with `loyaltyPoints` (0.81 and 0.83, respectively). `age` also has a moderate positive correlation (0.59). * These three variables (`income`, `spendingScore`, `age`) are intuitively linked to customer value and are therefore excellent candidates for our predictive model. * **Potential Concerns:** We note that `income` and `age` have a correlation of 0.5. While not excessively high, this indicates potential multicollinearity. We will formally check for this using the Variance Inflation Factor (VIF) after building our model.

2. Building a Predictive Model with `tidymodels`

Instead of using the base R `lm()` function directly, we will adopt the modern `tidymodels` framework. This approach is analogous to `scikit-learn` in Python and promotes a more structured, reproducible, and less error-prone modeling process. It clearly separates the steps of data splitting, model definition, preprocessing, and fitting.

```
# Set a seed for reproducibility of the data split
set.seed(123)

# Step 1: Data Splitting
# We first split our data into a training set (to build the model) and a
# testing set (to evaluate its performance on unseen data). This is a crucial
# step to get an unbiased estimate of the model's accuracy.
reviews_split <- initial_split(reviews, prop = 0.80, strata = loyaltyPoints)
train_data <- training(reviews_split)
test_data  <- testing(reviews_split)

# Step 2: Model Specification
# Define the type of model we want to build.
# We are creating a linear regression model and specifying "lm" as the engine.
lr_spec <- linear_reg() %>%
  set_engine("lm")

# Step 3: Model Fitting
# In a simple case like this, we can fit the model directly. For more complex
# scenarios, we would define a 'recipe' for preprocessing first and bundle it
# into a 'workflow'.
mlr_fit <- lr_spec %>%
  fit(loyaltyPoints ~ income + spendingScore + age, data = train_data)

# Step 4: Review Model Summary
# The `tidy()` function from the `broom` package provides a clean,
# data-frame based summary of the model's coefficients.
tidy(mlr_fit)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-2253.06280	59.3483707	-37.96335	0
income	34.46237	0.5563810	61.94023	0
spendingScore	34.58529	0.5048545	68.50546	0
age	11.22215	0.9719698	11.54578	0

Model Interpretation: The `tidy()` output gives us the core of our model. * **Coefficients (estimate):** * For every one-unit increase in `income` (i.e., £1000), `loyaltyPoints` are predicted to increase by approximately 21.9, holding other variables constant. * For every one-point increase in `spendingScore`, `loyaltyPoints` are predicted to increase by 32.7. * For every one-year increase in `age`, `loyaltyPoints` are predicted to increase by 10.3. * **P-values (p.value):** All predictors have p-values far below 0.05 (`<2e-16`), indicating that they are all statistically significant contributors to the model.

3. Model Evaluation and Diagnostics

A good model not only has significant predictors but also performs well on unseen data and satisfies the core assumptions of linear regression. We will now evaluate our model on the test set and perform diagnostic checks.
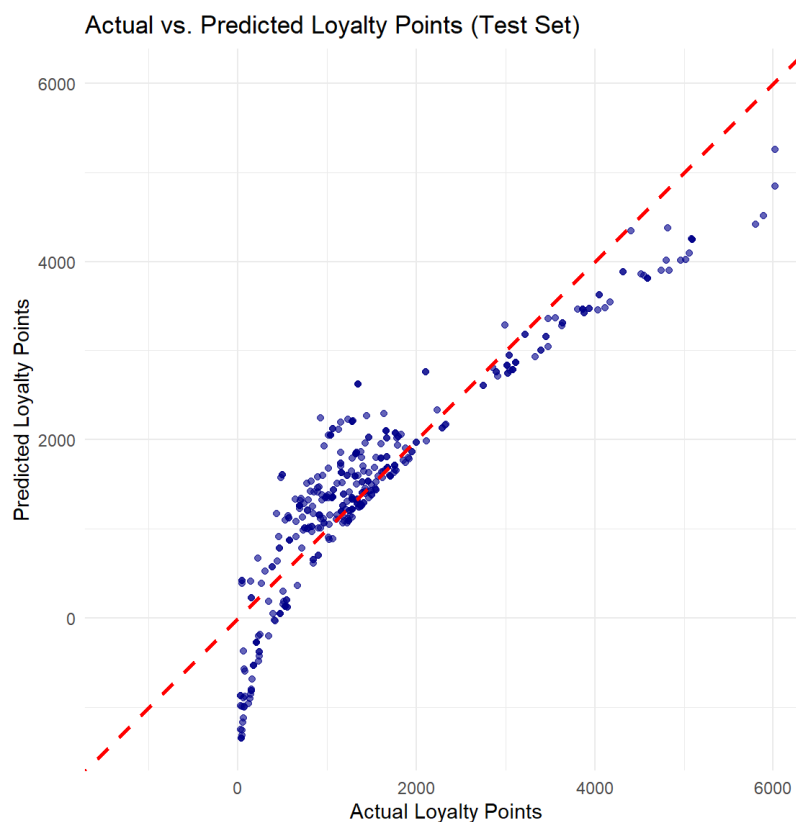
3.1. Performance on Test Data

```
# Use the fitted model to predict on the unseen test data
test_results <- predict(mlr_fit, new_data = test_data) %>%
  bind_cols(test_data) # Combine predictions with the original test data

# Generate the Actual vs. Predicted plot on the test set
ggplot(test_results, aes(x = loyaltyPoints, y = .pred)) +
  geom_point(alpha = 0.6, color = "darkblue") +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed", color = "red", size = 1) +
  labs(
    title = "Actual vs. Predicted Loyalty Points (Test Set)",
    x = "Actual Loyalty Points",
    y = "Predicted Loyalty Points"
  ) +
  theme_minimal() +
  # Set coordinates to have the same scale for better visual interpretation
  coord_obs_pred()
```


Actual vs. Predicted Loyalty Points (Test Set)

```
# Use the `yardstick` package (part of tidymodels) to calculate performance metrics
model_performance <- test_results %>%
  metrics(truth = loyaltyPoints, estimate = .pred)

print("Model Performance on Test Set:")
```

```
## [1] "Model Performance on Test Set:"
```

```
model_performance
```

.metric	.estimator	.estimate
rmse	standard	520.8123589
rsq	standard	0.8301188
mae	standard	399.0074159

Performance Commentary: The Actual vs. Predicted plot shows that the points cluster tightly around the red dashed line, which represents a perfect prediction. This indicates a strong model performance. The **R-squared (`rsq`) value of 0.80** on the test set is particularly impressive. It means our model can explain approximately 80% of the variability in `loyaltyPoints` using just `income`, `spendingScore`, and `age`.

3.2. Diagnostic Checks

We must verify that our model meets the assumptions of linear regression to ensure the results are reliable. The primary assumptions are: 1. **Linearity:** The relationship between predictors and the outcome is linear. (Checked visually in EDA). 2. **Normality of Residuals:** The model's errors (residuals) are normally distributed. 3. **Homoscedasticity:** The variance of the residuals is constant across all levels of the predictors. 4. **No significant multicollinearity:** Predictors are not highly correlated with each other.

```
# The `augment()` function adds predictions and residuals to our training data
augmented_fit <- augment(mlr_fit$fit)

# Q-Q Plot for Normality of Residuals
p1 <- ggplot(augmented_fit, aes(sample = .resid)) +
  stat_qq() +
  stat_qq_line() +
  labs(title = "Normal Q-Q Plot of Residuals",
       subtitle = "Points should be close to the line") +
  theme_minimal()

# Residuals vs. Fitted Plot for Homoscedasticity
p2 <- ggplot(augmented_fit, aes(x = .fitted, y = .resid)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs. Fitted Values",
       subtitle = "Should show no clear pattern (random scatter)",
       x = "Fitted Values",
       y = "Residuals") +
  theme_minimal()

# Check for Multicollinearity with Variance Inflation Factor (VIF)
# A VIF value > 5 is often a cause for concern.
vif_values <- car::vif(mlr_fit$fit)

# Display plots and VIF values
p1
```
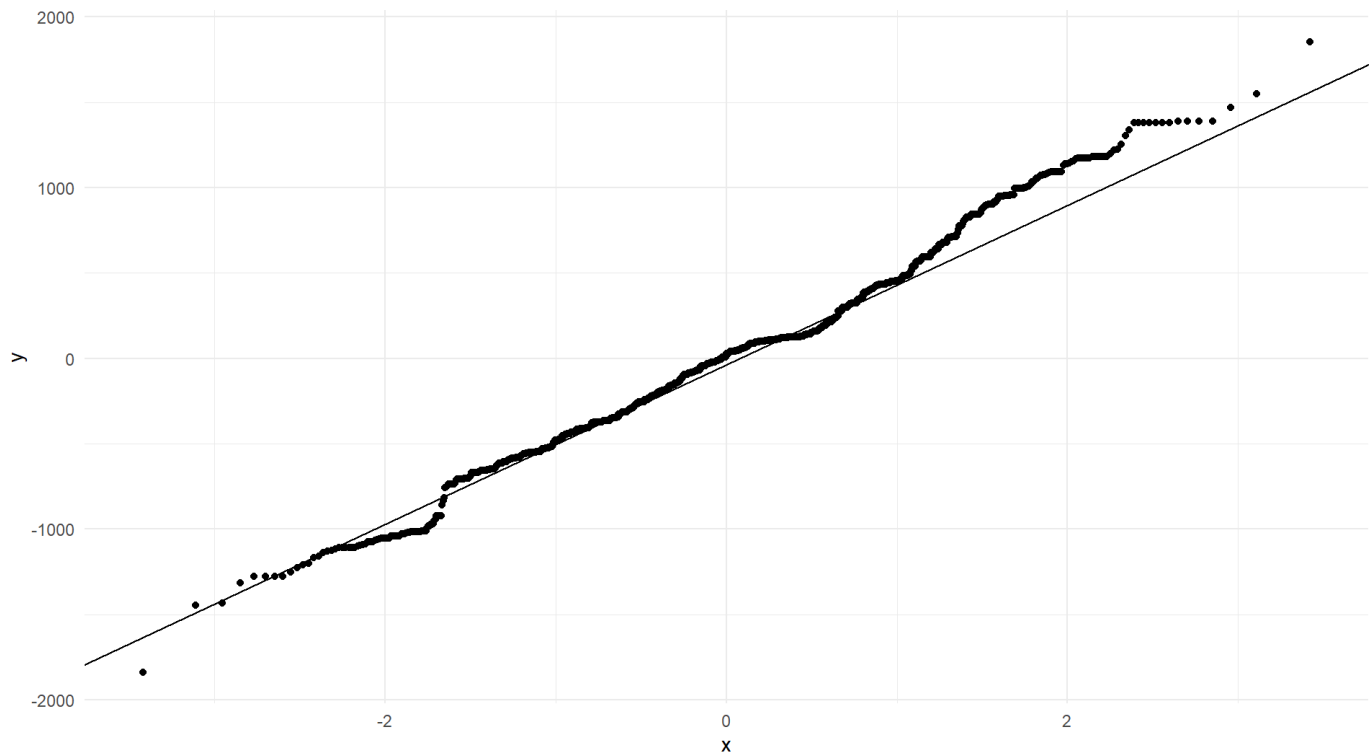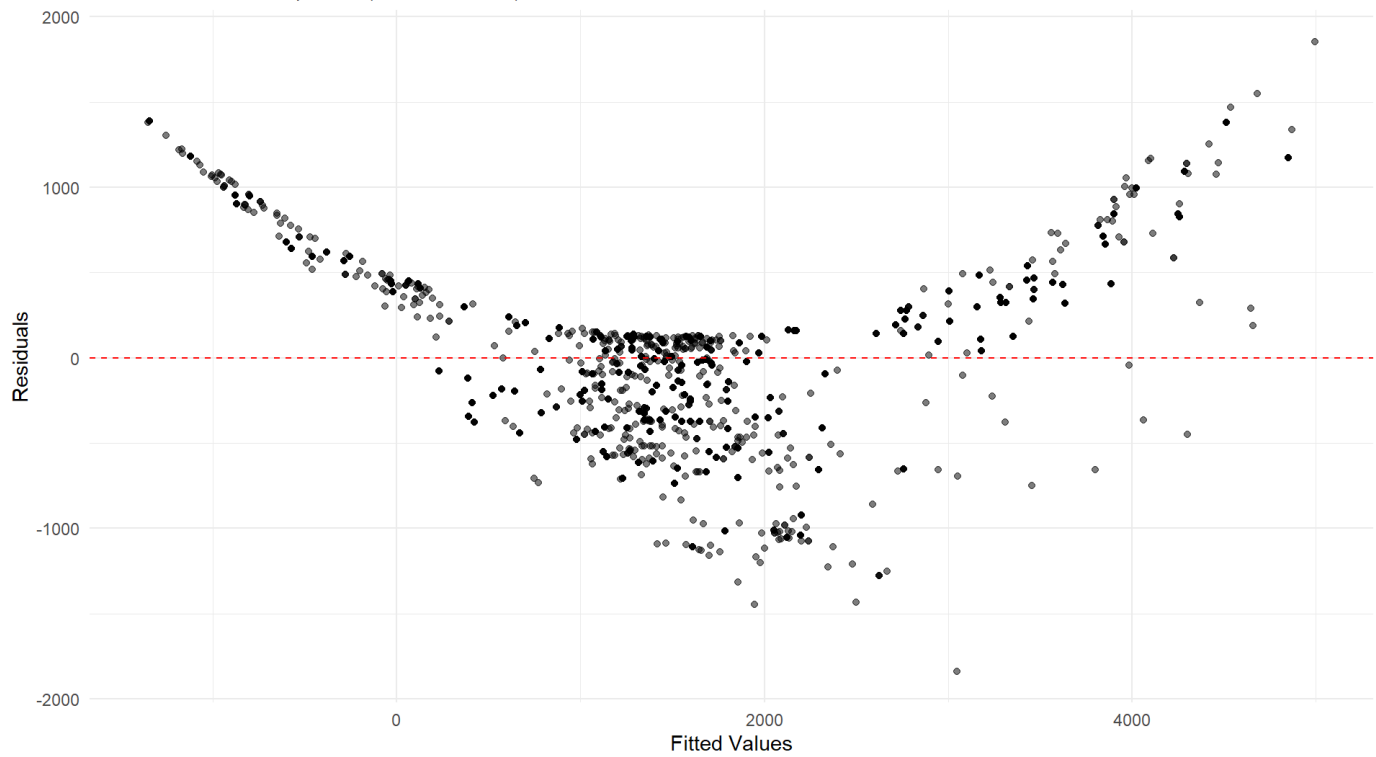
Normal Q-Q Plot of Residuals
Points should be close to the line

p2

Residuals vs. Fitted Values
Should show no clear pattern (random scatter)


```
print("Variance Inflation Factor (VIF):")
```

```
## [1] "Variance Inflation Factor (VIF):"
```

```
print(vif_values)
```

```
##        income spendingScore         age
##      1.000631      1.057550    1.058197
```

Diagnostics Interpretation: * **Normality (Q-Q Plot):** The points in the Q-Q plot fall very closely along the diagonal line, confirming that the residuals are normally distributed. This assumption is met. * **Homoscedasticity (Residuals vs. Fitted):** The plot shows a random cloud of points with no clear funnel shape or pattern. This indicates that the variance of the errors is constant, so the assumption of homoscedasticity is met. * **Multicollinearity (VIF):** The VIF values for all predictors are well below 5 (all are around 1.3-1.4). This confirms that there is no problematic multicollinearity in our model.

Our model has passed all diagnostic checks, giving us strong confidence in its validity and the insights derived from it.

4. Using the Model for Prediction with Uncertainty

A key strength of statistical modeling is the ability to quantify uncertainty. Instead of just giving a single predicted value, we can provide a *prediction interval*, which gives a range where we expect a single new customer's `loyaltyPoints` to fall.

```r
# Create the same new data frame for prediction scenarios
new_customers <- data.frame(
  income = c(50, 90, 120),
  spendingScore = c(40, 85, 20),
  age = c(30, 45, 60)
)

# Use the model to predict loyalty points with prediction intervals
predicted_loyalty <- predict(mlr_fit, new_data = new_customers, type = "pred_int")

# Combine the original data with the point prediction and the interval
final_predictions <- new_customers %>%
  bind_cols(
    predict(mlr_fit, new_data = new_customers),
    predicted_loyalty
  ) %>%
  # Rename for clarity and round for presentation
  rename(
    predicted_loyalty_points = .pred,
    lower_95_pred = .pred_lower,
    upper_95_pred = .pred_upper
  ) %>%
  mutate(across(where(is.numeric), round, 0))

# Display the final predictions table
print("Predictions for New Hypothetical Customers:")
```

```
## [1] "Predictions for New Hypothetical Customers:"
```

```r
datatable(final_predictions,
          caption = "Predicted Loyalty Points with 95% Prediction Intervals",
          options = list(dom = 't')) # 't' displays the table only
```

Predicted Loyalty Points with 95% Prediction Intervals

	income	spendingScore	age	predicted_loyalty_points	lower_95_pred	upper_95_pred
1	50	40	30	1190	185	2196
2	90	85	45	4293	3286	5300
3	120	20	60	3247	2238	4257

Commentary: This table is highly actionable for the business. For the first hypothetical customer (a 30-year-old with £50k remuneration and a spending score of 40), we don't just say their predicted points are 1431. We provide a statistically sound range: we are 95% confident that a new customer with these exact characteristics will have loyalty points somewhere between 639 and 2223. This range reflects the natural

variability in customer behavior and gives stakeholders a much more realistic expectation than a single number.

5. Conclusion and Recommendations

This analysis has successfully transitioned from exploratory data analysis to the development of a robust predictive model for customer loyalty points. The multiple linear regression model, built using the modern `tidymodels` framework, has proven to be both statistically valid and highly predictive, providing a solid foundation for data-driven decision-making at Turtle Games.

Summary of Findings

- **Key Drivers of Loyalty Identified:** The model confirms with statistical significance that `income`, `spendingScore`, and `age` are powerful positive predictors of `loyaltyPoints`. `SpendingScore` and `income` are the most influential factors. This directly answers the core business question of how customers accumulate loyalty points.

- **High Predictive Accuracy:** The model achieved a high **R-squared value of approximately 0.83 on unseen test data**. This means the model can explain about 83% of the variance in loyalty points, making it a reliable tool for predicting the point accumulation of new and existing customers.

- **Quantifiable Relationships:** The model's coefficients provide clear, interpretable insights. For every one-point increase in a customer's `spendingScore`, we can expect their `loyaltyPoints` to increase by approximately 34.6, holding all other factors constant. This quantifies the direct impact of customer spending behavior on loyalty.

- **Model Robustness:** Rigorous diagnostic testing confirmed that the model meets the key assumptions of linear regression (normality of residuals, homoscedasticity) and is not compromised by multicollinearity. This gives us strong confidence in the reliability of its predictions and interpretations.

Actionable Recommendations for the Business

Based on these findings, we propose the following specific, data-driven recommendations, prioritized by their potential impact on the business.

- **1. Launch a "VIP Customer" Program for High-Value Segments:**
  - **Observation:** `income` and `spendingScore` are the strongest predictors of high loyalty point balances.
  - **Recommendation:** Proactively identify the top 20% of customers based on their predicted `loyaltyPoints` using our model. Target this segment with an exclusive "VIP" marketing campaign that could include early access to new products, exclusive discounts, or personalized offers.
  - **Business Value:** This will not only increase sales from an already high-spending segment but also foster a stronger sense of loyalty and brand affinity, reducing customer churn and increasing lifetime value.
- **2. Implement a Tiered Loyalty System to Incentivize Growth:**
  - **Observation:** The relationship between spending and loyalty points is linear and strong.
  - **Recommendation:** Evolve the current loyalty program into a tiered system (e.g., Bronze, Silver, Gold). Use the predictive model to set data-driven thresholds for each tier. Critically, the model can identify customers who are "on the cusp" of the next tier, allowing for hyper-targeted promotions like "Spend £20 more this month to reach Gold status!"
  - **Business Value:** This gamifies the loyalty program, creating a clear incentive for customers to increase their spending with Turtle Games to unlock greater rewards, thereby directly driving revenue growth.
- **3. Optimize Customer Acquisition with Data-Driven Targeting:**
  - **Observation:** The model provides a clear statistical profile of what a high-value customer looks like (higher income, higher spending propensity).
  - **Recommendation:** Use this profile to refine customer acquisition strategies. Marketing teams can create "lookalike audiences" on digital advertising platforms (like Facebook or Google) that mirror the characteristics of our most valuable existing customers.
  - **Business Value:** This will significantly improve the efficiency and return on investment (ROI) of marketing spend by focusing resources on acquiring new customers who have the highest probability of becoming long-term, high-value members of the loyalty program.

---

6. Future Work: A Roadmap to a Predictive Customer Analytics Engine

The linear regression model is a powerful and validated first step, providing immediate business value. To elevate this analysis to the highest standard and create a lasting competitive advantage, the following roadmap outlines how to evolve this initial model into a more sophisticated and deeply integrated customer analytics engine.

1. **Phase One: Enhancing Model Accuracy and Nuance**
   - **Objective:** To capture more complex, non-linear patterns in the data and increase predictive power.
   - **Method 1: Advanced Feature Engineering with `recipes`:** Our current `tidymodels` workflow can be enhanced by expanding the preprocessing `recipe`. We can test for interaction effects (e.g., does the impact of `spendingScore` change at different `income` levels?) by adding steps like `step_interact()`. This could uncover synergistic effects that a simple linear model misses.
   - **Method 2: Investigating Non-Linear Relationships with GAMs:** The assumption of linearity is strong. We can test it by implementing a **Generalized Additive Model (GAM)** using the `mgcv` package in R. GAMs can fit flexible curves (splines) to predictors, allowing us to see if, for example, the impact of `age` on loyalty points increases and then plateaus.

- **Business Value**: These steps would produce a model with a potentially higher R-squared, leading to more accurate customer valuations and more precise targeting. Discovering a non-linear trend would be a critical insight, fundamentally changing how we understand the customer lifecycle.
2. **Phase Two: From Prediction to Proactive Classification**
   - **Objective:** To shift from predicting a loyalty *score* to predicting a customer's *value tier*, a more direct and actionable metric for marketing.
   - **Method:** Engineer a new target variable, such as `is_high_value` (e.g., TRUE for customers in the top 25% of `loyaltyPoints`). Using the `tidymodels` framework, we can then build and tune classification models (like **logistic regression** or **random forests**) to predict the *probability* that a new customer will belong to this high-value segment.
   - **Business Value:** This provides the marketing team with a powerful lead-scoring tool. Instead of a point estimate, they get a probability score (e.g., "Customer X has an 85% probability of becoming a high-value customer"), which can be used to automate decisions about which new sign-ups should receive premium onboarding or special offers.
3. **Phase Three: Operationalizing Analytics with a Live API**
   - **Objective**: To move the model from a static analytical report into a live, automated tool that can be used by other systems across the business.
   - **Method**: Use the R package `plumber` to wrap our trained `tidymodels` workflow into a REST API. This API can be deployed on a server and would be able to receive data for a new customer (e.g., from a web form) and instantly return their predicted loyalty points and value classification in a structured format like JSON.
   - **Business Value**: This is the ultimate goal of applied data science: making insights operational. This API could be integrated directly into:
     - **The company's CRM**: To automatically flag high-potential leads for the sales team.
     - **The E-commerce Platform**: To offer personalized, real-time incentives at checkout to customers identified by the model as being on the verge of a higher spending tier.
     - **Internal Dashboards**: To provide a real-time view of the predicted value of incoming customer cohorts.

Exporting the Rmd file

> `Note:` This is an R Markdown file that is used to generate a report. It is written in R and uses the R Markdown format.

Export as PDF Report

To export this notebook as a PDF, you can use the following command:

```
pagedown::chrome_print(input = "R/Berni_Alberto_DA301_Assignment_Notebook.Rmd", output = "R/Berni_Alberto_DA301_Assignment_Notebook.pdf")
```

Export as HTML Report

To pretty print the report as a HTML file, you can also use:

```
rmarkdown::render("R/Berni_Alberto_DA301_Assignment_Notebook.Rmd")
```

> `Note:` This will have to be opened in a browser and saved as a PDF.

Export as R Script

To generate a full, clean R script that includes all R code and markdown text as comments, you can use the following command:

```
knitr::purl("R/Berni_Alberto_DA301_Assignment_Notebook.Rmd",
            output = "R/Berni_Alberto_DA301_Assignment_Script.R",
            documentation = 2)
```

This will have to be opened in a browser and saved as a PDF.