Turtle Games Analysis – Technical Report.

Background and Context:

Turtle Games is a game manufacturer and distributor that is looking to improve sales performance through the analysis of customers' trends and reviews. To generate meaningful and actionable insights, it wants to look more closely at three questions:

- How do customers accumulate/engage with loyalty points?
    - And what does this insight tell us about possible predictive models?
- What possible groups can be individuated for targeted marketing campaigns?
- How can customers reviews help inform market campaigns and the business?

Analytical Approach:

Regression Models:

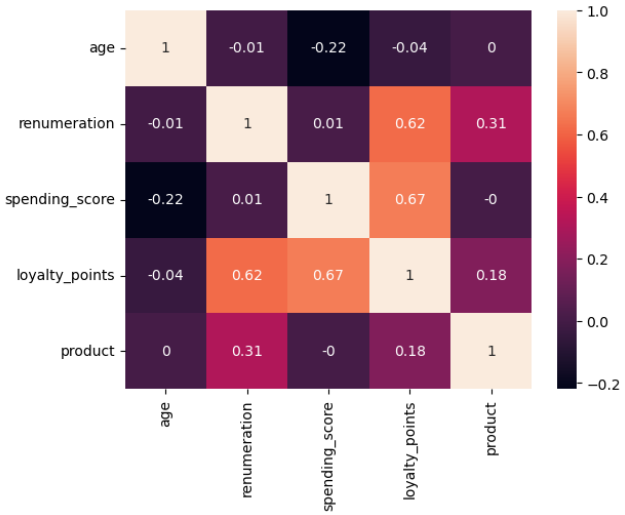
The first steps in the analytical process is to clean and explore the data in an iterative manner. The cleaning process includes:

- Removing redundant columns and renaming others.
- Checking for duplicates.
- Checking for missing values.

In this phase, I looked into the various Turtle games' demographics to get acquainted with the data and audience. Preliminary observations:

- There is a slight majority of female customers.
- The average age is 39.5.

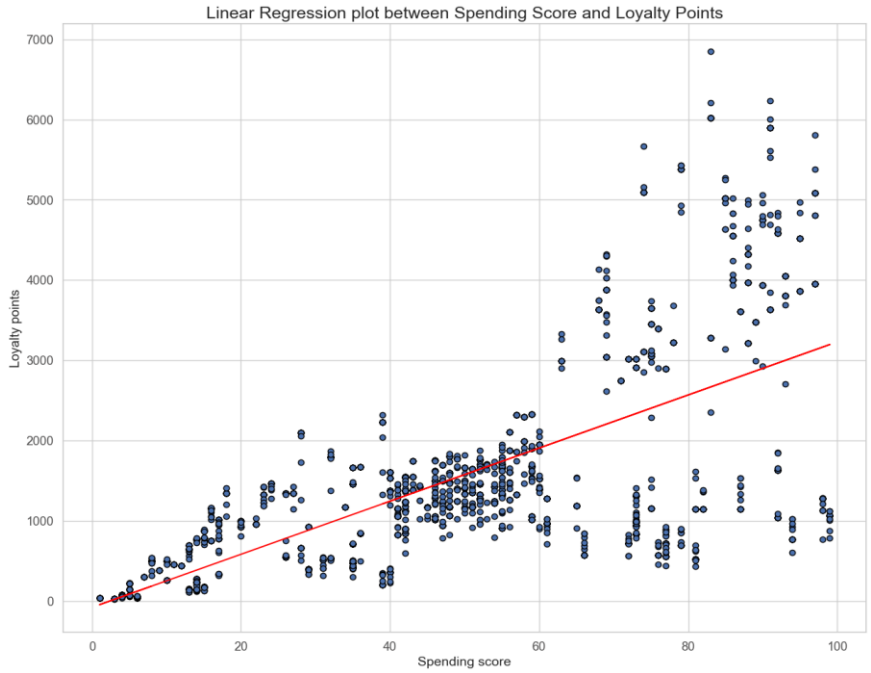
Following the initial exploration, I investigated the possible correlations between the dependent variable ('loyalty_points') and independent variables. The correlation matrix showed that 'remuneration' and 'spending_score' had a significant positive correlation score to 'loyalty_points' and, 'age', had a slightly negative one. Those are the three variables the analysis focused on.

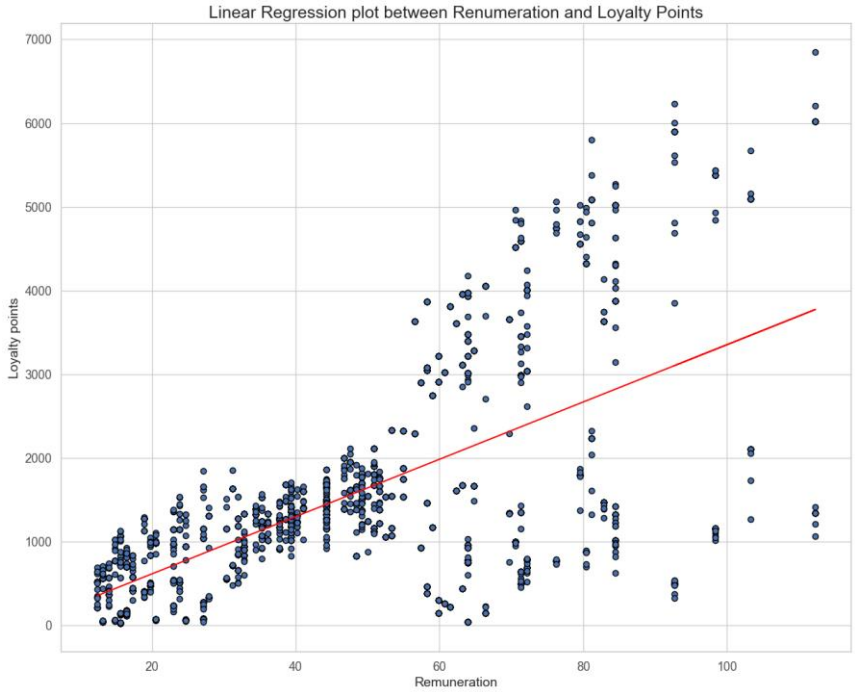


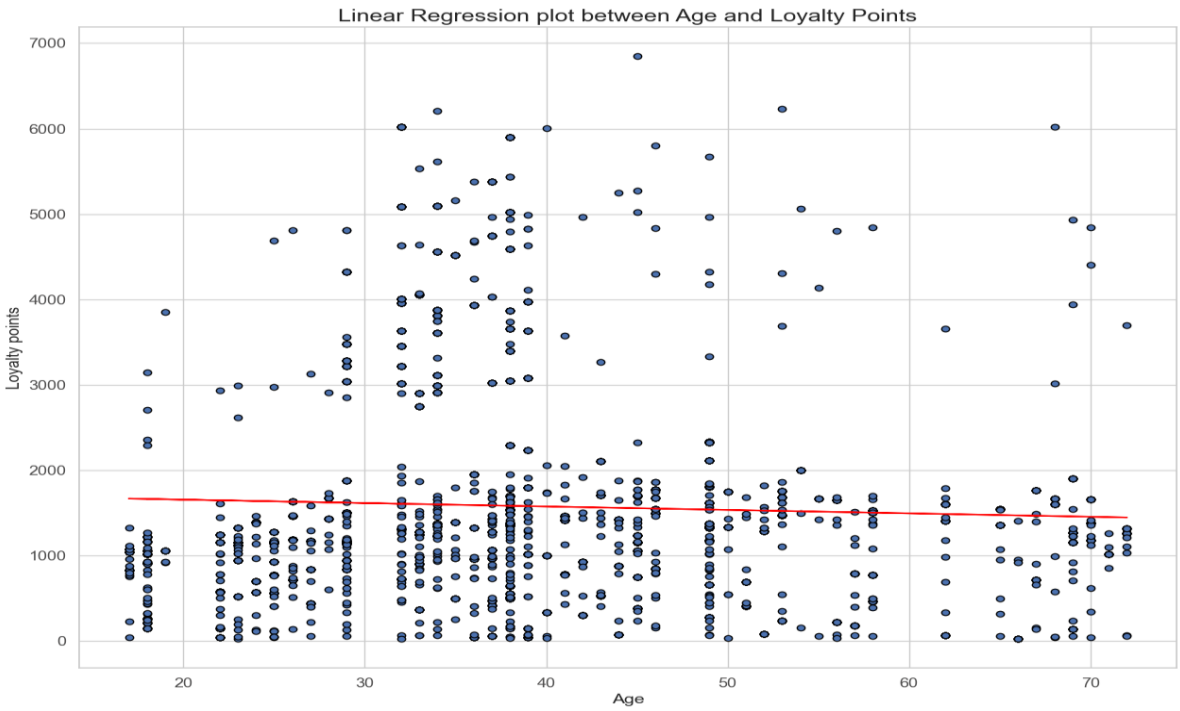
I began with univariate linear regression model to individually examine the relationship between the independent variable and the 'loyalty_points'. The models showed that each variable can explain, in a limited way, the variability of the dependent variable:

- Model 1 ('spending_score') showed an R-squared of 45%.

- Model 2 (‘remuneration’) showed an R-squared of 38%.
- Model 3 (‘age) showed an R-squared of 2%.







Model 3 showed a negative regression line between 'age' and 'loyalty points'.

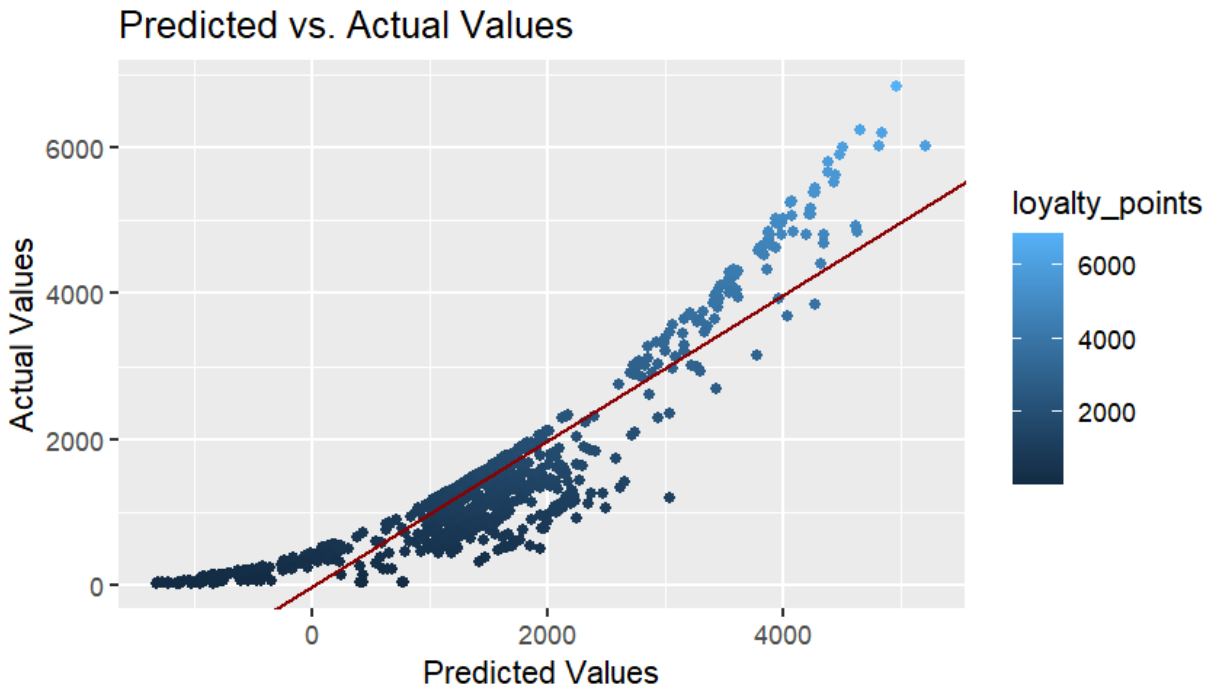
To have a more robust predictive model, I combined the independent variables to build a multiple linear regression model. In Python, I excluded the 'age' variable to not compromise linearity, while in R, it was kept as it showed significance. The multiple linear regression model was built by splitting the data in train and test sets and fitted into the OLS model, the results showed:

- A R-squared value of 0.821, meaning it explains 82% of the variability in 'loyalty_points'.
- A low t-value of 0, meaning both variables are significant.

The model was then checked to verify the assumptions needed:

- Through the VIF, the model was checked for multicollinearity and with a score of 1 for both variables, it confirmed no correlation between the variables.
- By plotting the residuals against the predicted values, no cone shape is visible, the Breausch-Pagan method (p-value above 0.05) confirmed homoscedasticity.
- Through a histogram and Q-Qplot the model confirmed to have a normal distribution for the residuals, although slightly skewed to the left.
- Through the Darbin-Watson method, I confirmed the assumption of independence with a value of 1.97 and 2 as the ideal result, there is no serial correlation.

In R, the multiple regression model was not checked against its assumptions but plotted to understand how its predictions compared to the actual values. By adding 'age' into the model, it also showed a slight improvement in the R-squared value being roughly 84%.

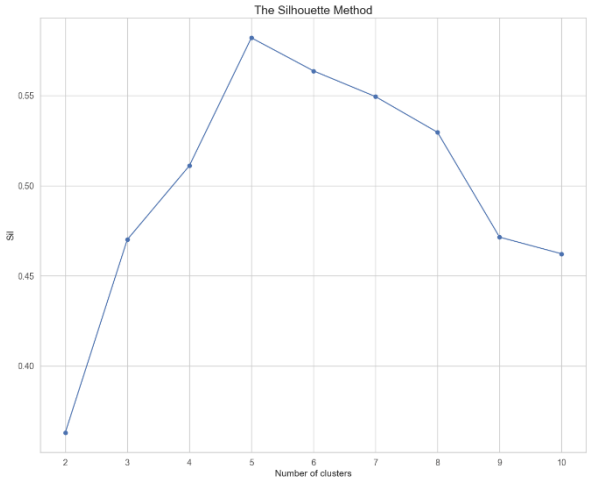


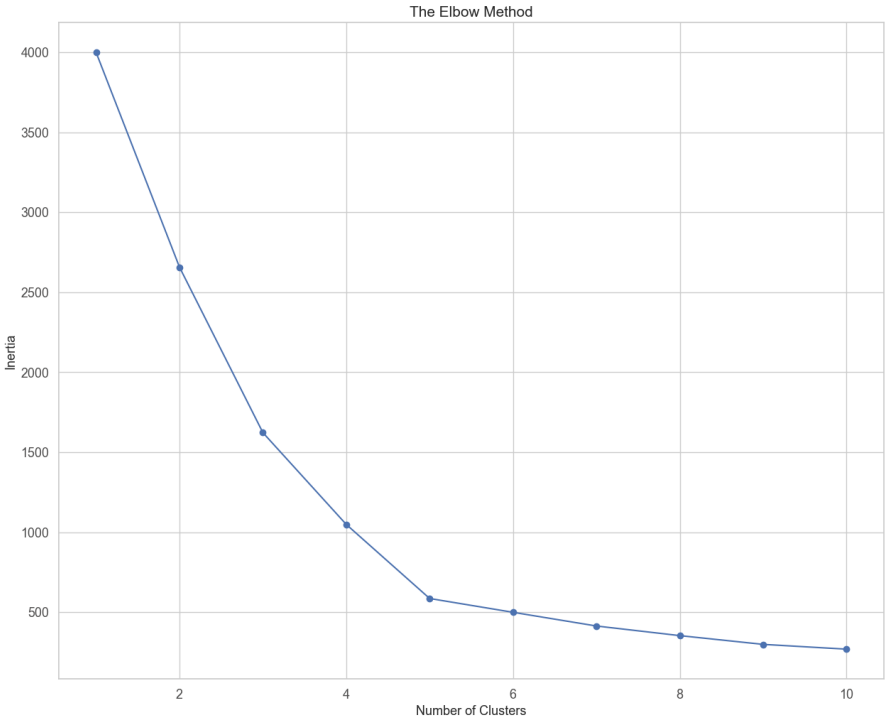
The graph shows that the predicted values are fairly conservative to the actual values. Considering its predictions and its R-squared values the model can be improved but is a good starting point in understanding how loyalty points work.

A Decision Tree Regressor was built as well but as, the model showed a low accuracy score, it was disregarded from analysis.

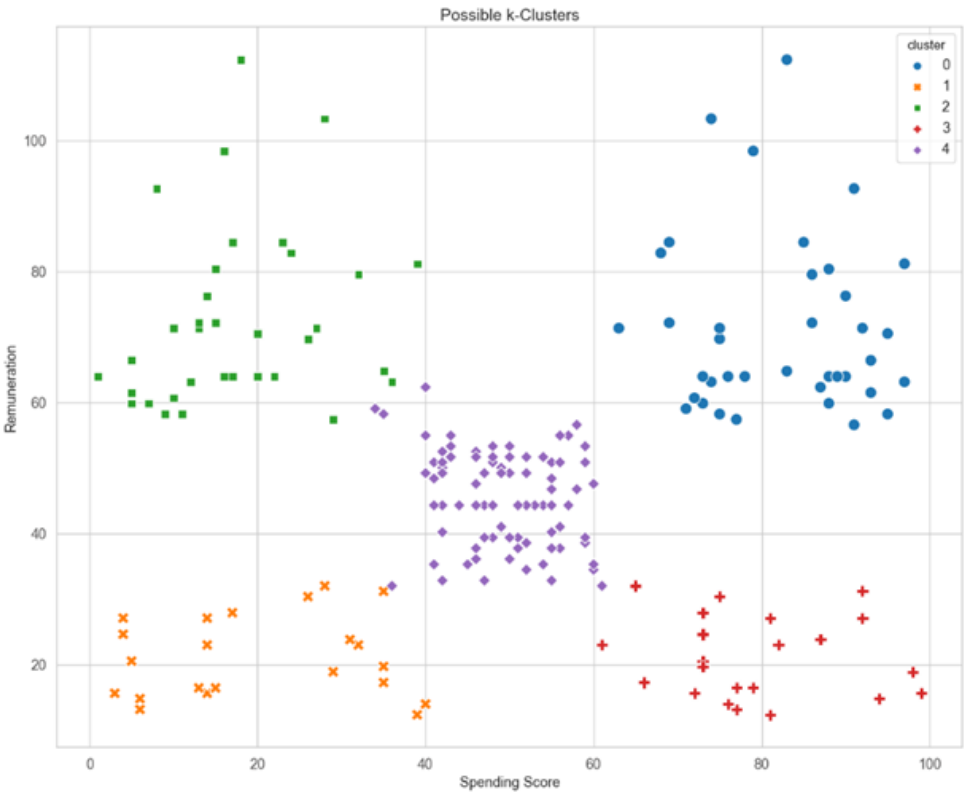
Clusters and Targeted Campaigns:

With the significance of 'remuneration' and 'spending score' in understanding the accumulation of 'loyalty points', the analysis aimed to understand possible customers' behaviours related to, how they spend, and how much they earn to individuate groups for targeted marketing campaigns. To find possible classifications, I employed the Elbow and Silhouette method to find clusters in the relationship between the two variables.





Both showed an ideal number of 5 k-clusters that from the scatter point I could group into the following categories:



- Bottom left: low spending score, low remuneration.
- Bottom right: low spending score, high remuneration.
- Center: medium spending score, medium remuneration.
- Top left: high spending score, low remuneration.
- Top right: high spending score, high remuneration.

By breaking the customers behaviours in different groups, we can observe that big spenders have similar behaviour and the same can be said for small spenders:

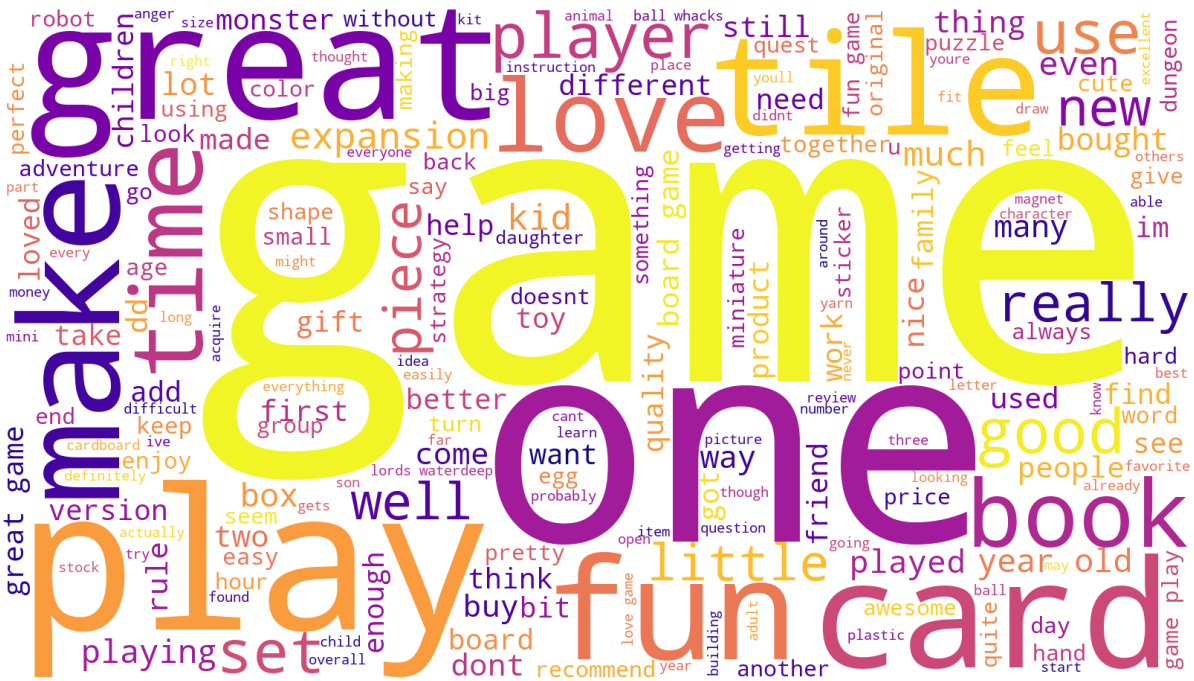
- Age is a factor as shown by the high spenders, low remuneration having a younger demographic vs low spenders having an older average age.
- The most populated category is medium spenders with a medium remuneration, which is the backbone of the company's earnings and customer base.

From these 5 clusters, we can inform marketing campaigns that are suitable for each group:

- Group 1 (low spending / low earnings) is not of great interest as it's one of the smallest categories and does not generate much revenue.
- Group 2 (high spending / low earning) is more high-risk as it tends to spend more irresponsibly, as such promotions and reduced offers can be a significant way to attract both Group 1 and 2 and generate more revenue.
- Group 3 (medium spending and earnings) is the most populated group and should not be alienated at the cost of losing a solid client base.
- Group 4 (low spending / high earnings) can be persuaded into promotion and reduced offers to gain loyalty and revenue.
- Group 5 (high spenders and earners) should be targeted in bonus offers and promotions to generate the biggest sales.

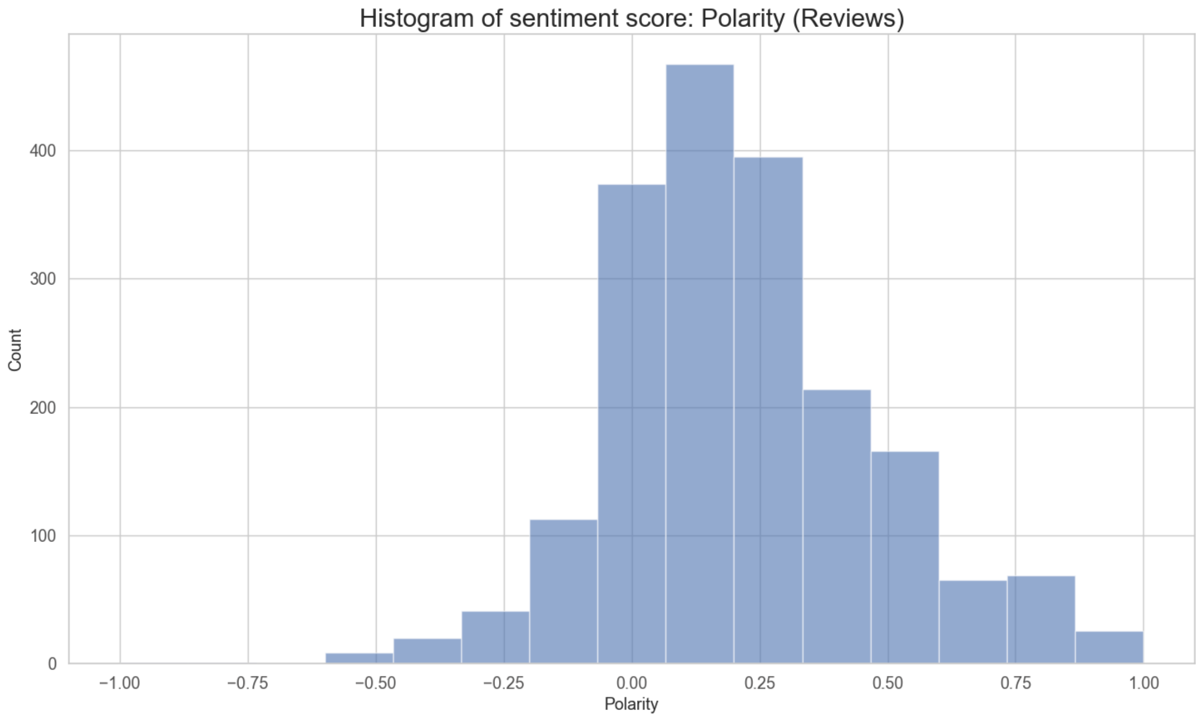
Customer Reviews Analysis:

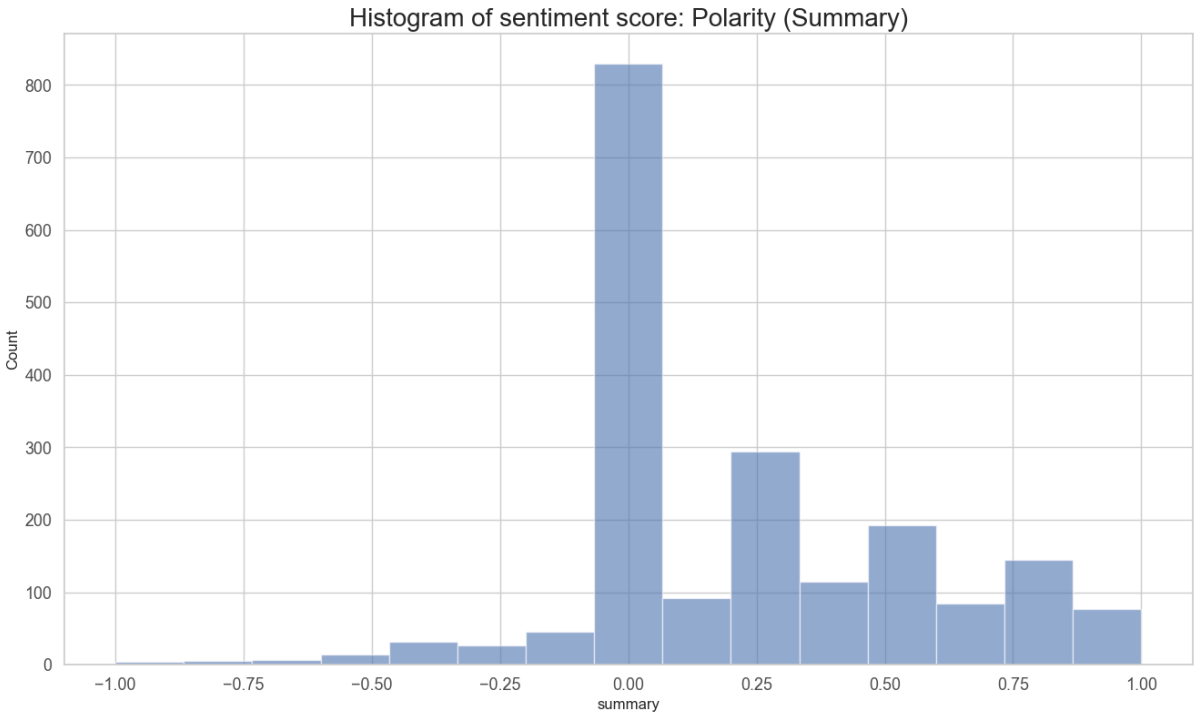
To launch more pointed marketing campaigns, it is important to understand the general sentiment of the customer base. Having cleaned data, I tokenised the words and removed the stopwords generating graphs of the most frequent words:



The word clouds give us a glance of the overall sentiment which appears positive with words like ‘fun’, ‘great’, ‘five’ and ‘stars’.

Performing a polarity and subjectivity analysis we can confirm a general positive tendency, with both the review and summary column having a strong neutrality but skewness to the right as well as a bigger kurtosis on the positive side.





From looking at the most negative comments we see a prevalence of ‘boring’ or ‘difficult’ ,from the positive a more frequent use of the word ‘awesome’. This gives us indication that some games need to be slightly more accessible and leaning into fun.

Conclusion and Insights:

- The linear regression model showed an importance of 84% in explaining how loyalty points work. As spending score, remuneration and age grow, so do loyalty points, but the slight uncertainty of the model means it should be used with caution or be built upon.
- The classification technique showed 5 main group of customer behaviours linked to spending score and remuneration. To optimise marketing campaigns and revenues, it can focus on flash deals and reduced offers for small spenders, loyalty programs for medium ones, and bonus offers for big spenders.
- The customer review analysis showed a general positive sentiment in the client base. It can be enhanced by leaning more into fun and ease.

Appendix

Data cleaning in Python:

```
# Any missing values?
# Replace the missing values with 0.
reviews.fillna(0, inplace=True)

# Determine the number of missing values.
reviews.isnull().sum()
```

```
gender                    0
age                       0
remuneration (k£)         0
spending_score (1-100)    0
loyalty_points            0
education                 0
language                  0
platform                  0
product                   0
review                    0
summary                   0
dtype: int64
```

```
# Explore the data.
reviews.drop_duplicates()
print (reviews.shape)
print(reviews.value_counts())
```

```
# Basic descriptive statistics.
reviews.describe()
```

	age	remuneration (k£)	spending_score (1-100)	loyalty_points	product
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	39.495000	48.079060	50.000000	1578.032000	4320.521500
std	13.573212	23.123984	26.094702	1283.239705	3148.938839
min	17.000000	12.300000	1.000000	25.000000	107.000000
25%	29.000000	30.340000	32.000000	772.000000	1589.250000
50%	38.000000	47.150000	50.000000	1276.000000	3624.000000
75%	49.000000	63.960000	73.000000	1751.250000	6654.000000
max	72.000000	112.340000	99.000000	6847.000000	11086.000000

```
# Drop unnecessary columns.
reviews.drop(['language', 'platform'], axis=1, inplace=True)

# View column names.
reviews.columns
```

```
Index(['gender', 'age', 'remuneration (k£)', 'spending_score (1-100)',
       'loyalty_points', 'education', 'product', 'review', 'summary'],
      dtype='object')
```

In R:

```r
# Read file and summary.
rev <- read.csv('turtle_reviews.csv', header = TRUE)
df_rev <- subset(rev, select = -c(language, platform))
colnames(df_rev)[3] <- 'remuneration'
colnames(df_rev)[4] <- 'spending_score'
```

Multiple Linear Regression:

```python
# Create independent variable containing all three previous variables.
x4 = reviews_clean[['renumeration', 'spending_score']]

# Split the data in 'train' (80%) and 'test' (20%) sets.
X_train, X_test, Y_train, Y_test = train_test_split(x4, y, test_size=0.20, random_state=5)

# Add a constant.
X_train = sm.add_constant(X_train)

# Training the model using the 'statsmodel' OLS library.
# Fit the model with the added constant.
model4 = sm.OLS(Y_train, X_train).fit()
model4.summary()
```

Checking Assumptions:

```python
# Checking for multicollinearity.
# Add a constant.
x_temp = sm.add_constant(X_train)

# Create an empty DataFrame.
vif = pd.DataFrame()

# Calculate the 'vif' for each value.
vif['VIF Factor'] = [variance_inflation_factor(x_temp.values, i)
                     for i in range(x_temp.values.shape[1])]

# Create the feature columns.
vif['features'] = x_temp.columns

# Print the values to two decimal points.
print(vif.round(2))
```

```
   VIF Factor        features
0        9.45           const
1        1.00    renumeration
2        1.00  spending_score
```

```
# check normality of residuals
from scipy.stats import shapiro
#perform Shapiro-Wilk test
shapiro(residuals)
```

```
ShapiroResult(statistic=0.9912799000740051, pvalue=3.703021178580457e-08)
```

The residuals seems fairly normally distributed. While the histogram is not a perfect bell shape it does follow a close pattern. The Q-Q plot showed more clearly how the residuals are distributed along a normal distribution line. similarly, we confirm that the distribution is normal through the Shapiro-Wilk test where we reject the null hypotesis if the p-value is less than 0.05: with a p-value of 3.70 we fail to reject the null hypotesis and we confirm normal distribuition.

```
# Checking for homoscedasticity with the Breusch Pagan method.
import statsmodels.stats.api as sms
test = sms.het_breuschpagan(model4.resid, model4.model.exog)
terms = ['LM stat', 'LM Test p-value', 'F-stat', 'F-test p-value']
print(dict(zip(terms, test)))
```

```
{'LM stat': 39.230974380834205, 'LM Test p-value': 3.0276254927307376e-09, 'F-stat': 20.070832089116433, 'F-test p-value': 2.46
09115607488977e-09}
```

Both the graphical rapresentation and the Breusch Pagan method confirm that homoscedasticity in the model as the p-value is greater than 0.05 for the Breusch-Pagan results and the graph doesn't show a cone like pattern.

```
# Checking for independence.
# The Durbin-Watson method.
from statsmodels.stats.stattools import durbin_watson
durbin_watson(residuals)
```

```
1.9704874859603527
```

The Durbin-Watson method is based on a scale between 0 and 4, with a result closer to 0 meaning a positive serial correlation and with a result closer to 4 meaning a negative serial correlation, and with 2 meaning no correlation. With a result of 1.970 we can confirm our assumption of independence.

Data cleaning for NLP examples:

```
# Review: Change all to lower case and join with a space.
df_nlp['review'] = df_nlp['review'].apply(lambda x: " ".join(x.lower() for x in x.split()))
df_nlp
```

	review	summary
0	when it comes to a dm's screen, the space on t...	The fact that 50% of this space is wasted on a...
1	an open letter to galeforce9*: your unpainted ...	Another worthless Dungeon Master's screen from...
2	nice art, nice printing. why two panels are fi...	pretty, but also pretty useless
3	amazing buy! bought it as a gift for our new d...	Five Stars
4	as my review of gf9's previous screens these w...	Money trap
...	...	...
1995	the perfect word game for mixed ages (with mom...	The perfect word game for mixed ages (with Mom
1996	great game. did not think i would like it when...	Super fun
1997	great game for all..... keeps the mind nim...	Great Game
1998	fun game!	Four Stars
1999	this game is fun. a lot like scrabble without ...	Love this game

2000 rows × 2 columns

```
# Replace all the punctuations in review column.
# function to remove punctuation:
def remove_punct(text):
    punctfree="".join([i for i in text if i not in string.punctuation])
    return punctfree

# Updating review column.
df_nlp['review'] = df_nlp['review'].apply(lambda x:remove_punct(x))

df_nlp
```

	review	summary
0	when it comes to a dms screen the space on the...	the fact that 50% of this space is wasted on a...
1	an open letter to galeforce9 your unpainted mi...	another worthless dungeon master's screen from...
2	nice art nice printing why two panels are fill...	pretty, but also pretty useless
3	amazing buy bought it as a gift for our new dm...	five stars
4	as my review of gf9s previous screens these we...	money trap
...	...	...