**PROBABILITY AND STATISTICAL INFERENCE FOR DATA SCIENCE**

# PREDICTING GAMER CHURN

## A Deep Learning and Statistical Analysis of Gaming Behavior

Sri Ranga Jayanth Jammula (sj1149)

Deviram Kondaveti (dk1273)

Deepak Sai Are (da910)

Vaishnavi Madhavaram (vm695)

# Table of Contents

# 1. INTRODUCTION

Churn refers to the phenomenon where players discontinue their engagement with a game over a period of time. In the gaming industry, churn represents a critical challenge, as it directly impacts revenue, player base sustainability, and the overall success of gaming platforms. Players who churn may do so due to various reasons, including loss of interest, frustration with gameplay, or external factors like competing games. The gaming industry operates in a highly competitive environment, with new titles and updates constantly vying for players' attention. High churn rates translate to significant revenue losses, as acquiring new players is substantially costlier than retaining existing ones. Moreover, churn negatively affects community dynamics within games, leading to reduced engagement among active players. The ability to predict and address churn is vital for gaming companies to:

- Develop targeted retention strategies.
- Personalize user experiences.
- Enhance game design to better align with player preferences.
- Sustain long-term growth of their platforms.

## 1.1. Objective of the Study

This project aims to analyze player behavior to predict churn using advanced deep learning techniques and statistical methods. By identifying patterns and key engagement metrics, the study seeks to provide actionable insights that can help gaming companies:

- Improve player retention.
- Enhance the overall gaming experience.
- Minimize revenue loss through informed intervention strategies.

# 2. DATA COLLECTION & PREPROCESSING

## 2.1. DATA COLLECTION

The match data for League of Legends was collected using the *matchdata_batch.py* script. This script retrieves detailed player behavior and gameplay data from the Riot Production API (We were able to secure riots production API for which we had to fill in an application as this contained player specific match data which is sensitive information), leveraging its comprehensive coverage of match statistics and player engagement metrics.

1. **Seed Player-Based Retrieval:** The data collection began by using a seed player's PUUID (a player's unique identifier) to fetch their match history.
2. **Iterative Expansion:** Competitor PUUIDs (opponents and teammates) were extracted from each match's participant list. This iterative process enabled the collection of a broader dataset, encompassing diverse player behaviors.
3. **Data Scope:** Over 7,000 players were sampled, with up to 100 matches per player, ensuring extensive coverage of player activity.

4. **Features Collected:**

   The dataset retrieved includes 144 features spanning match metadata, player performance metrics, and engagement indicators critical for churn prediction:

   - **Match Metadata**: Includes game details like gameId, gameMode, gameDuration, gameType, and timestamps (gameCreation, gameStartTimestamp, gameEndTimestamp).
   - **Player Performance Metrics:** Captures detailed combat stats such as kills, assists, deaths, goldEarned, goldSpent, visionScore, turretTakedowns, and inhibitorTakedowns.
   - **Engagement Metrics:** Tracks wardsPlaced, timeCCingOthers, longestTimeSpentLiving, totalDamageDealt, and behavioral data like teamPosition.
   - **Outcome Indicators:** Includes results such as win (True/False) and endOfGameResult.

   These features provide a comprehensive view of player behavior and match outcomes, forming the foundation for accurate churn prediction.

5. **Storage:** The collected data was structured and saved in both CSV and JSON formats for subsequent preprocessing and modeling.

## 2.2. DATA CLEANING

The data cleaning process was meticulously performed to ensure the dataset's quality and reliability for subsequent analysis and modeling. Key steps included:

1. **Handling Missing Values:**
   Missing values in critical columns such as total_game_duration, win_loss_ratio, and kill_death_ratio were identified and addressed. If a column was missing entirely, it was flagged for further review. For partially missing values:
   - Imputation methods like mean or median filling were applied.
   - Records with a significant amount of missing data in key features were removed.
2. **Duplicate Removal:**
   To maintain data integrity and prevent bias, duplicate entries in the dataset were identified and removed.
3. **Data Formatting:**
   - Timestamps such as gameCreation, gameStartTimestamp, and gameEndTimestamp were converted to human-readable datetime formats.
   - Converted outcomes and flags into numeric formats for analysis, ensuring compatibility with machine learning models.

This allowed for time-based analyses such as session durations and intervals between matches. These steps ensured a clean, reliable, and robust dataset ready for further feature engineering and modeling.

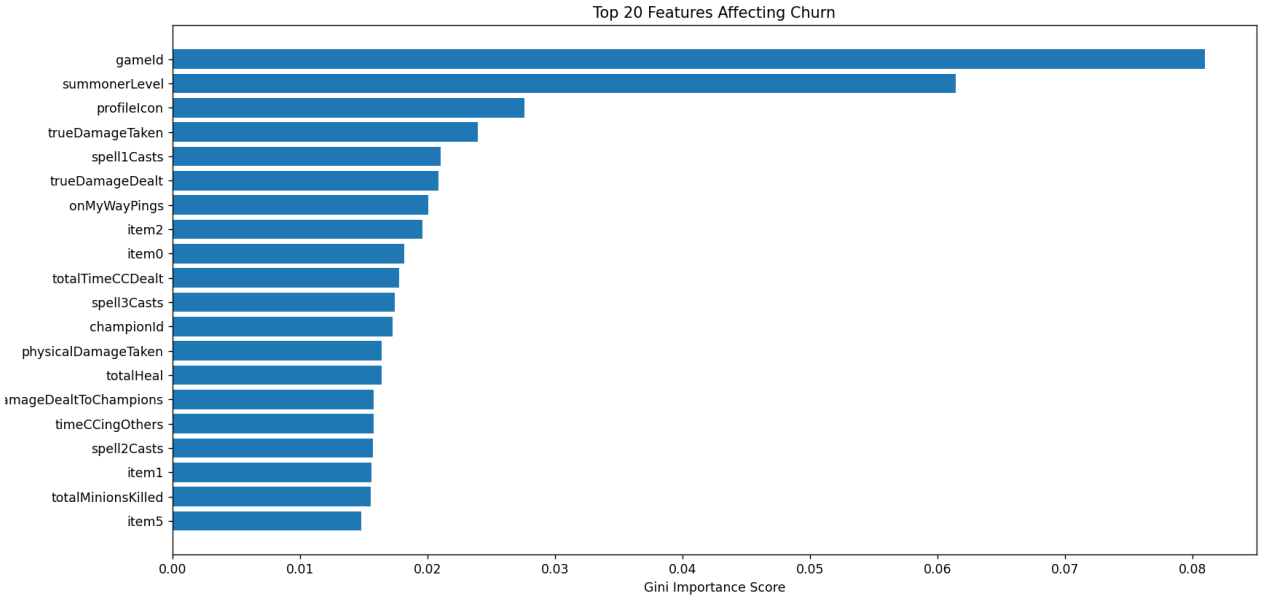
# 3. FEATURE EXTRACTION

This section outlines the methodology used to extract and engineer features for predicting player churn. The process involved several iterations of feature selection and aggregation, eventually leading to a refined set of meaningful features. The following sections detail the initial approach, the challenges faced, the refined approach, and the final feature set.

To identify the most relevant features, we initially applied different feature importance metrics. However, the results were not as expected. Here is the approach we followed, and the challenges encountered.

## 3.1. GINI IMPORTANCE

- We used Gini importance to rank features, but it highlighted features like gameID, profileIcon, championID, and timeCCing as important. Intuitively, these features were unlikely to significantly affect player churn, so we deemed this approach ineffective.



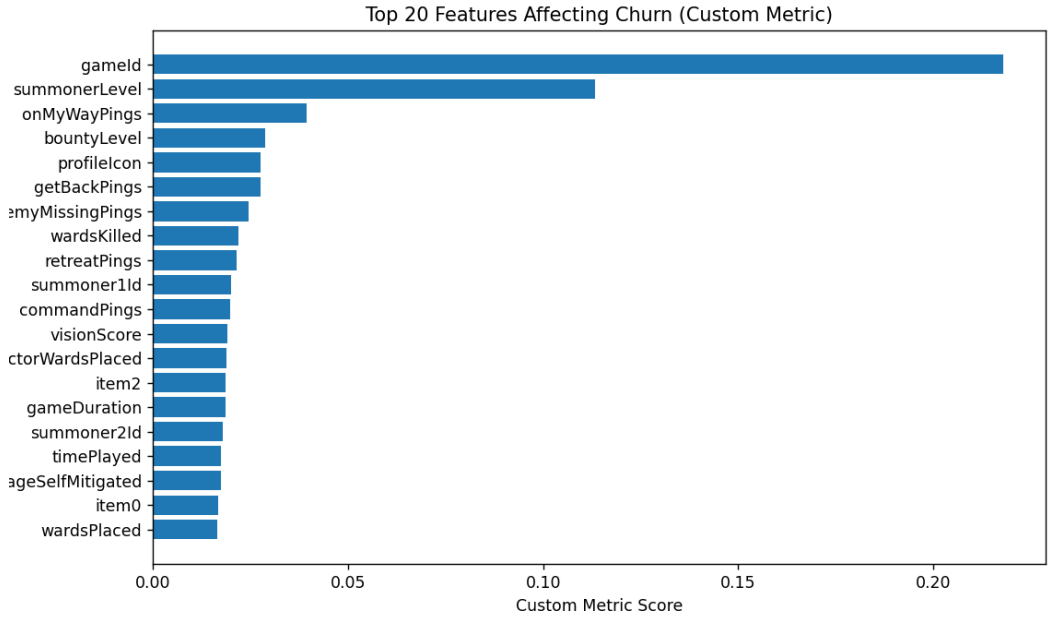

**Custom Feature Importance Metric**

- To improve upon Gini importance, we developed a custom feature importance score. This score was calculated as a weighted combination of three metrics:

**Custom Metric Score** = α * Entropy Value + β * Pearson Correlation + Γ * Gini Importance

  - **Entropy Value**: Measured the unpredictability or randomness of a feature.
  - **Pearson Correlation**: Captured linear relationships between the feature and the churn label.
  - **Gini Importance**: Used to account for the impurity reduction in decision tree-based models.

- We utilized a GridCV with a time-series split to compute these metrics for each feature and assigned weights (α, β, Γ) to combine them into a single score.




**Challenges Faced**

- Despite creating this more robust approach, many of the top-ranked features still did not align with logical expectations. For instance, non-informative features like gameID and profileIcon appeared as important.

## 3.2. REFINED SEARCH

Given the challenges with automated selection methods; we shifted to a more logical, domain-driven approach to select features. This approach aimed to select features that logically influence player churn. We reduced the initial set of 114 features to the following 9 core features.

**Selected Features (Per Game)**

1. **PUUID**: Unique player identifier.
2. **Total Game Duration**: Duration of the match.
3. **Time Between Games**: Time gap between consecutive games.
4. **Win**: Whether the player won the game (binary feature).
5. **Unique Champions**: Total number of unique champions used by the player.
6. **Kills**: Number of kills achieved in the game.
7. **Deaths**: Number of times the player died in the game.
8. **Champion Level**: Level achieved by the player's champion in the match.
9. **Gold Usage**: Total gold spent during the game.
10. **Churn**: Target variable indicating player churn.

## 3.3. FEATURE AGGREGATION

Since each player can have up to 100 games, we aggregated features across these games to create player-level features. Aggregation involved averaging or summarizing features to capture player behavior. The following transformations were applied:

**Aggregated Features**

1. **PUUID**: Player identifier (unchanged).
2. **Average Game Duration**: Calculated as total game duration divided by the total number of games for the player.

$$avg\_game\_duration = \frac{total\_game\_duration\_per\_match}{total\_games}$$

3. **Average Time Between Games**: Calculated as the time between the end of one game and the start of the next, averaged across all games for the player.

$$avg\_time\_between\_games = \frac{game\_end\_timestamp\_current\_game \; - \; game\_start\_timestamp\_next\_game}{number\_of\_gaps}$$

4. **Win-Loss Ratio**: Ratio of wins to losses for the player.

$$win\_loss\_ratio \; = \frac{number\_of\_wins}{number\_of\_losses}$$

5. **Unique Champions**: Total number of distinct champions used by the player.

$$unique\_champions \; = \; count(unique(champion\_id))$$

6. **Kill-Death Ratio**: Ratio of total kills to total deaths for the player.

$$kill\_death\_ratio \; = \frac{total\_kills}{total\_deaths}$$

7. **Average Champion Level**: Average of the champion level achieved across all games for the player.

$$avg\_champion\_level = \frac{total\_champion\_level}{otal\_games}$$

8. **Average Gold Usage**: Total gold spent by the player divided by the total number of games.

$$avg\_gold\_usage \; = \frac{total\_gold\_usage}{total\_games}$$

9. **Average Daily Streak**: Average number of games played on consecutive days.

$$avg\_daily\_streak = \frac{total\_consecutive\_play\_days}{total\_active\_days}$$

10. **Total Active Days**: The total number of days in which the player was active.

$$total\_active\_days = len(dates\_played)$$

11. **Churn**: Final target variable representing whether a player churned.

### 3.4. SUMMARY OF IMPROVEMENTS

1. **Logical Selection**: Shifted from purely automated selection to a logical, domain-driven process.
2. **Feature Aggregation**: Aggregated game-level features to player-level features, enhancing predictive power.
3. **Reduced Dimensionality**: Reduced the 114 initial features to 11 meaningful and interpretable features.
4. **Improved Intuition**: Selected features that are directly linked to player engagement and churn (e.g., win-loss ratio, average time between games, and average daily streak).
5. **Churn Definition**: Clearly defined churn as the target variable, which was missing in the initial automated feature importance calculations.

This refined methodology improved interpretability and predictive accuracy by focusing on logically significant features and leveraging domain expertise in player behavior. The final player-level feature set is more compact, understandable, and aligned with the goal of predicting player churn.

## 4. EXPLORATORY DATA ANALYSIS

The Exploratory Data Analysis (EDA) stage is critical for uncovering patterns, trends, and relationships within the dataset, which inform feature selection and model development.

### 4.1. DESCRIPTIVE STATISTICS

Descriptive statistics were used to summarize and understand the central tendencies, variability, and distributions of the dataset. Key metrics such as kill_death_ratio, avg_gold_usage, and unique_champions were analyzed to provide a foundation for further analysis.

| | total_game_duration | win_loss_ratio | unique_champions | \ |
|---|---|---|---|---|
| count | 6899.000000 | 6899.000000 | 6899.000000 | |
| mean | 1603.767894 | 0.831593 | 1.890709 | |
| std | 355.758660 | 0.909287 | 2.023979 | |
| min | 103.000000 | 0.000000 | 1.000000 | |
| 25% | 1418.583333 | 0.000000 | 1.000000 | |
| 50% | 1616.000000 | 1.000000 | 1.000000 | |
| 75% | 1810.000000 | 1.000000 | 2.000000 | |
| max | 3051.000000 | 13.000000 | 78.000000 | |

| | kill_death_ratio | avg_champion_level | avg_gold_usage | avg_daily_streak |
|---|---|---|---|---|
| count | 6899.000000 | 6899.000000 | 6899.000000 | 6899.000000 |
| mean | 1.293563 | 13.667432 | 20162.317204 | 2.186694 |
| std | 1.599623 | 2.567478 | 6266.684677 | 2.690402 |
| min | 0.000000 | 1.000000 | 500.000000 | 1.000000 |
| 25% | 0.444444 | 12.000000 | 15914.375000 | 1.000000 |
| 50% | 0.894737 | 14.000000 | 20059.583333 | 1.000000 |
| 75% | 1.500000 | 15.200000 | 24095.170455 | 2.000000 |
| max | 21.000000 | 18.000000 | 52579.000000 | 114.000000 |

| | total_active_days |
|---|---|
| count | 6899.000000 |
| mean | 2.186694 |
| std | 2.690402 |
| min | 1.000000 |
| 25% | 1.000000 |
| 50% | 1.000000 |
| 75% | 2.000000 |
| max | 114.000000 |

**Analysis of Key Features:**

This table summarizes 8 numeric features with key statistics, including mean, standard deviation, minimum, percentiles**,** and maximum. Below is the concise analysis for each feature based on these descriptive statistics:

1. **total_game_duration:** Average match duration is ~26.7 minutes, with longer games (up to ~50.8 minutes) showing balanced engagement but potential fatigue risks.
2. **win_loss_ratio:** Players generally win slightly more than they lose (mean = 0.83), with extreme values suggesting outliers or skilled players.
3. **unique_champions**: Most players use 1-2 champions, while some experiment with up to 78, reflecting diverse gameplay strategies.
4. **kill_death_ratio:** Average is 1.29, showing slightly more kills than deaths, with high outliers indicating exceptional performance.
5. **avg_champion_level:** Players typically reach level 13.67, indicating consistent progression; lower levels may correlate with disengagement.
6. **avg_gold_usage:** Mean gold usage is 20,162, with higher values reflecting better in-game performance and engagement.
7. **avg_daily_streak:** Most players have short streaks (~2 days), while longer streaks (up to 114 days) indicate loyalty and high activity.
8. **total_active_days:** Players are active for an average of 2.18 days, but dedicated players show extended engagement (up to 114 days).

These insights help identify player behavior patterns and inform strategies for churn prevention and engagement optimization.

**Key Insights:**

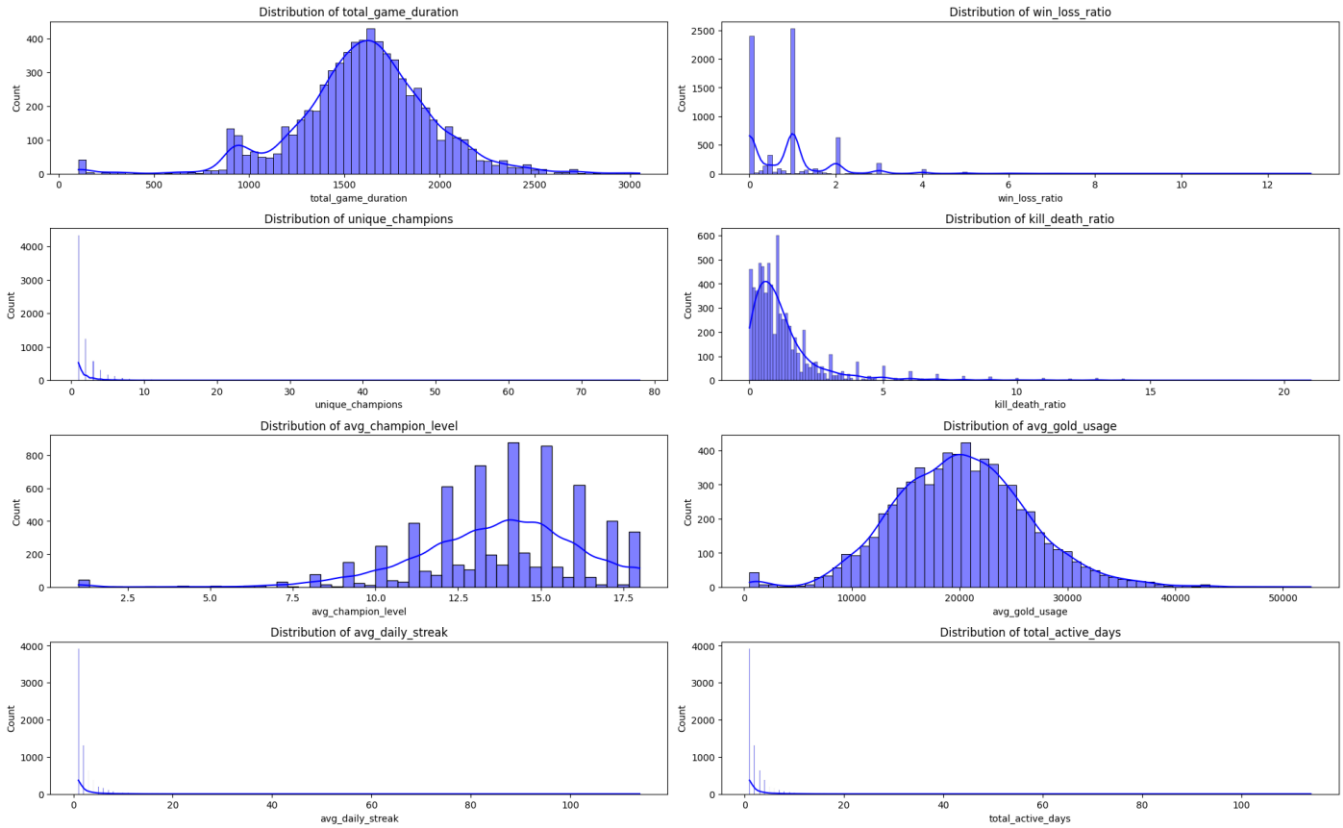
- Engagement Levels: Players with higher avg_daily_streak and total_active_days are likely more invested, while low engagement metrics suggest at-risk players for churn.
- Performance Indicators: High variability in kill_death_ratio and avg_gold_usage suggests differences in player skill and gameplay styles.
- Outliers: Extremely high values for unique_champions, win_loss_ratio, and kill_death_ratio may need further investigation as potential outliers.

## 4.2. VISUALIZATIONS

We used the visualizations in the project to provide insights into the dataset's distribution, relationships, and potential anomalies.

### A. Histograms

To visualize the distribution of each feature and identify patterns, such as normal distributions, skewness, or multimodality.




Histograms are used to visualize the distribution of key metrics in the dataset. Here's a breakdown of insights derived from the provided histograms:

1. **total_game_duration:**
    - The distribution is slightly right-skewed, with most matches lasting between 1,400 and 1,800 seconds (~23 to 30 minutes).

- A small number of longer matches (up to 3,000 seconds or ~50 minutes) suggest unique gameplay scenarios that may reflect high engagement or prolonged strategies.

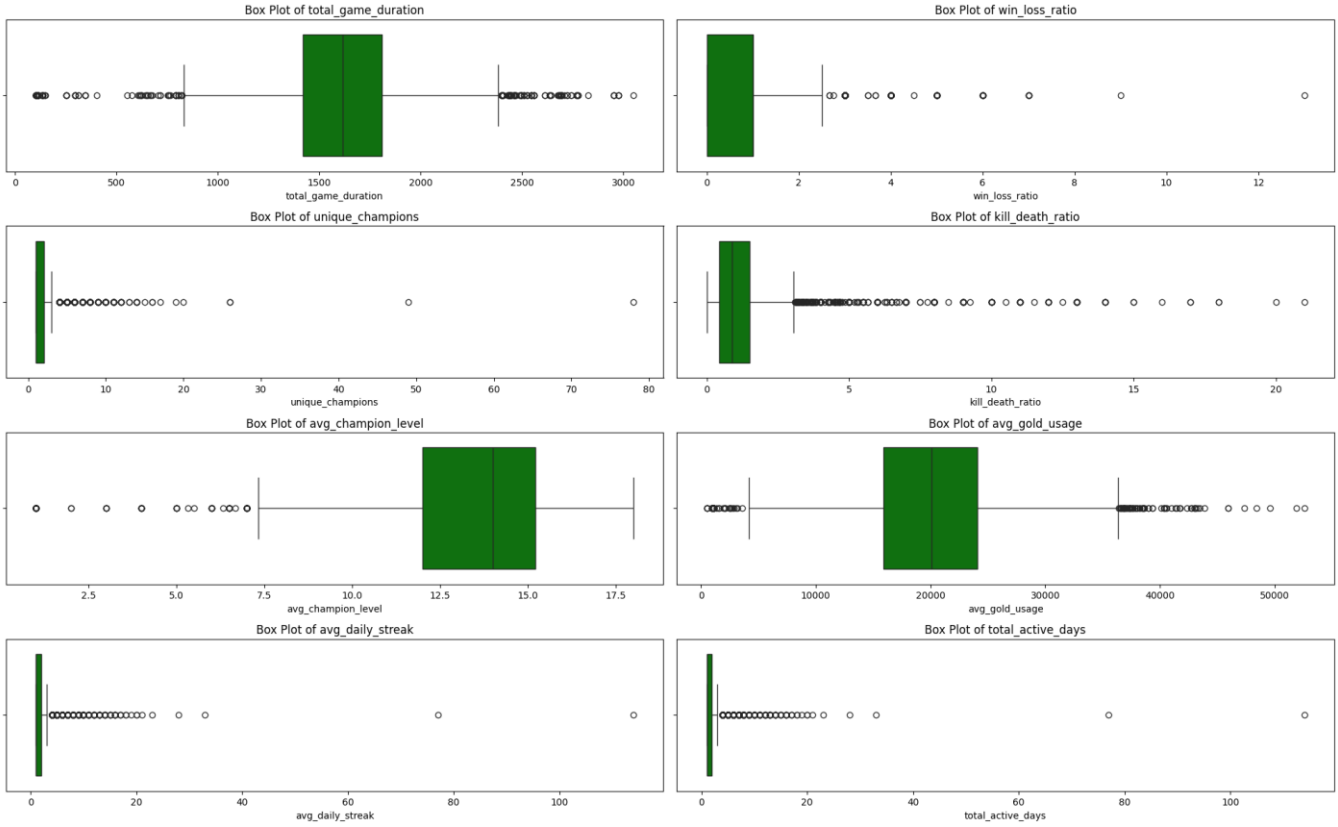
2. **win_loss_ratio:**
   - The distribution is heavily skewed to the left, with a large portion of players clustered around 0 and 1, indicating a balanced or slightly favorable win-loss performance for most.
   - Outliers with ratios above 5 may represent highly skilled players or data anomalies.
3. **unique_champions:**
   - The histogram shows that the majority of players use 1-2 champions, with a sharp decline in frequency as the number of unique champions increases.
   - A few outliers who use up to 78 champions may indicate experimentation or mastery across multiple roles.
4. **kill_death_ratio:**
   - The distribution is highly skewed to the left, with most players having a ratio between 0 and 2.
   - A few players achieve extremely high kill/death ratios (up to 20), reflecting exceptional performance or rare circumstances.
5. **avg_champion_level:**
   - The distribution is bell-shaped, with most players reaching levels 12 to 15 during matches, indicating consistent in-game progression.
   - A smaller portion reaches level 18, representing peak in-game achievement.
6. **avg_gold_usage:**
   - The distribution is roughly normal, centered around 20,000 gold usage.
   - A tail extending to 50,000 suggests a few players accumulate and utilize significantly more gold, likely in longer or more competitive matches.

**Key Insights:**

- **Player Performance:** Metrics like kill_death_ratio and avg_gold_usage reveal variability in player skill and engagement levels. Outliers in these distributions may require further investigation.
- **Player Engagement:** Metrics such as total_game_duration and unique_champions highlight differences in how players engage with the game, with most exhibiting standard behavior and a few deviating significantly.
- **Targeted Interventions:** The histograms suggest opportunities for improving engagement and retention, such as addressing challenges faced by players with low win/loss ratios or encouraging diverse champion usage.

## B. Box Plots

Box Plots are used to detect outliers and understand feature variability in the data.




1. **total_game_duration:**
   - The interquartile range (IQR) indicates most games last between 1500-2000 seconds (~25-33 minutes).
   - Outliers on the higher end suggest unusually long game sessions, possibly reflecting marathon gaming or anomalous data.
2. **win_loss_ratio:**
   - The IQR is tightly clustered around a value less than 1, indicating balanced gameplay but a slight lean toward losses.
   - The high outliers (values >4) could represent highly skilled or lucky players.
3. **unique_champions:**
   - A significant concentration around lower values (1-10), with few players experimenting with more than 50 champions.
   - The extreme outlier (close to 80 champions) might indicate an exceptionally diverse player.
4. **kill_death_ratio:**
   - Majority of players maintain a ratio near 1, suggesting balanced gameplay.
   - Outliers extending above 10 indicate highly skilled or exceptional players.

5. **avg_champion_level:**
    - Most players achieve champion levels between 12 and 16, indicating consistent mid-game performance.
    - Outliers below 7 might signify disengaged or new players struggling to progress.
6. **avg_gold_usage:**
    - IQR indicates standard gold usage between 10,000 and 30,000, aligning with typical in-game performance.
    - Outliers above 40,000 could reflect extended matches or highly efficient players.
7. **avg_daily_streak:**
    - Most players engage for short streaks (less than 10 days), with outliers suggesting streaks of over 40 days, indicating strong engagement or loyalty.
8. **total_active_days:**
    - Similar to avg_daily_streak, the majority of players are active for fewer than 10 days.
    - A few players are active for over 100 days, reflecting high retention and loyalty.

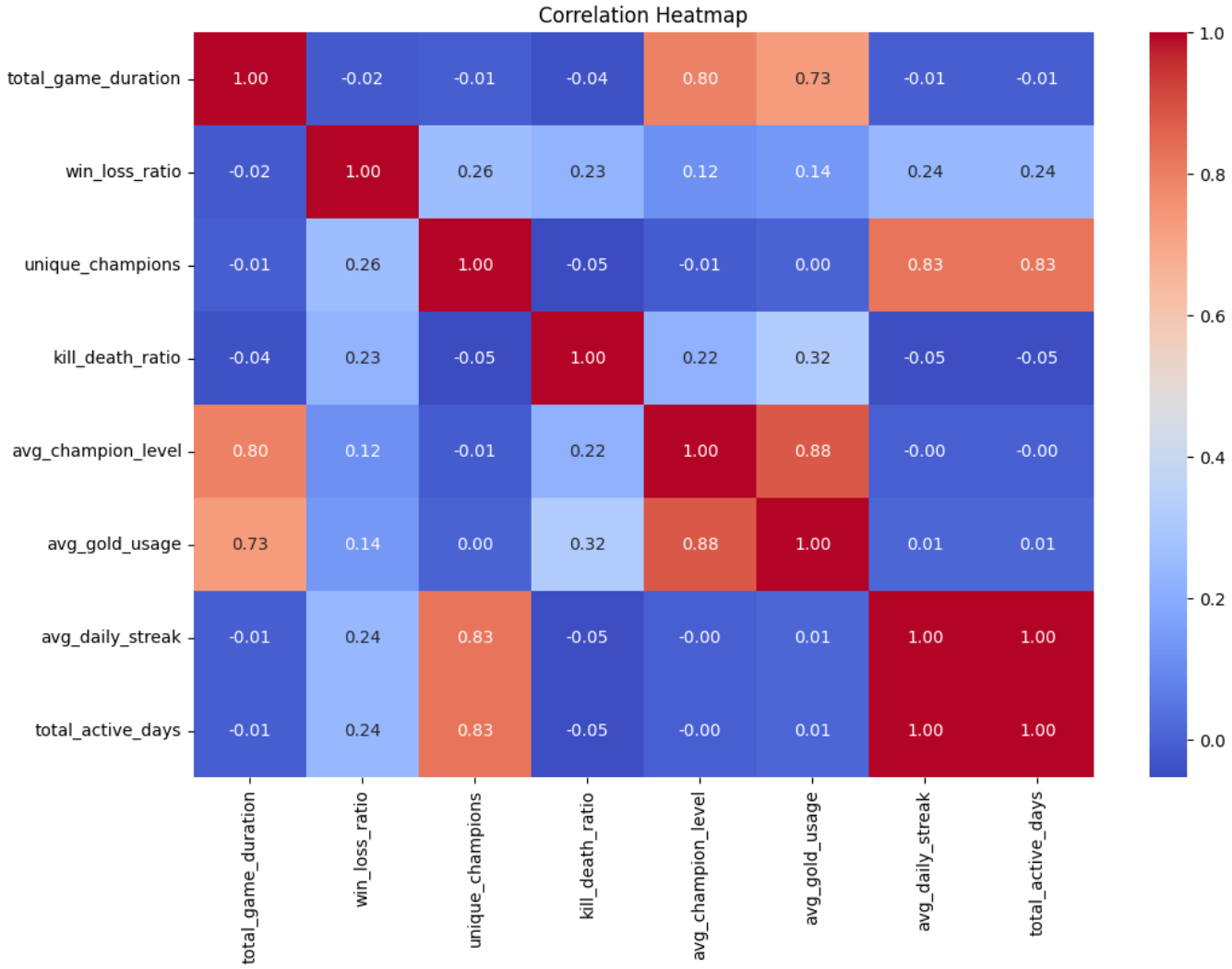
**Key Insights:**

- **Engagement vs. Disengagement:** Players with lower avg_champion_level, kill_death_ratio, or avg_gold_usage might face difficulty in gameplay, risking churn. Players with higher metrics in these areas exhibit higher engagement and loyalty.
- **Targeted Interventions:** Players with high avg_daily_streak and total_active_days should be rewarded to sustain loyalty. Identify and support players in the lower IQRs of avg_champion_level and win_loss_ratio to reduce churn.
- **Outlier Management:** High outliers in kill_death_ratio and unique_champions could skew analyses; consider robust statistical techniques or capping to minimize their influence.

### C. Correlation Heatmaps

Correlation maps were used to examine relationships between features and detect multicollinearity. This heatmap informs data preprocessing steps, feature engineering, and the focus areas for deeper analysis.

The correlation heatmap provides a visual representation of the relationships between numeric features in the dataset. Each cell shows the correlation coefficient (ranging from -1 to 1) between two variables.

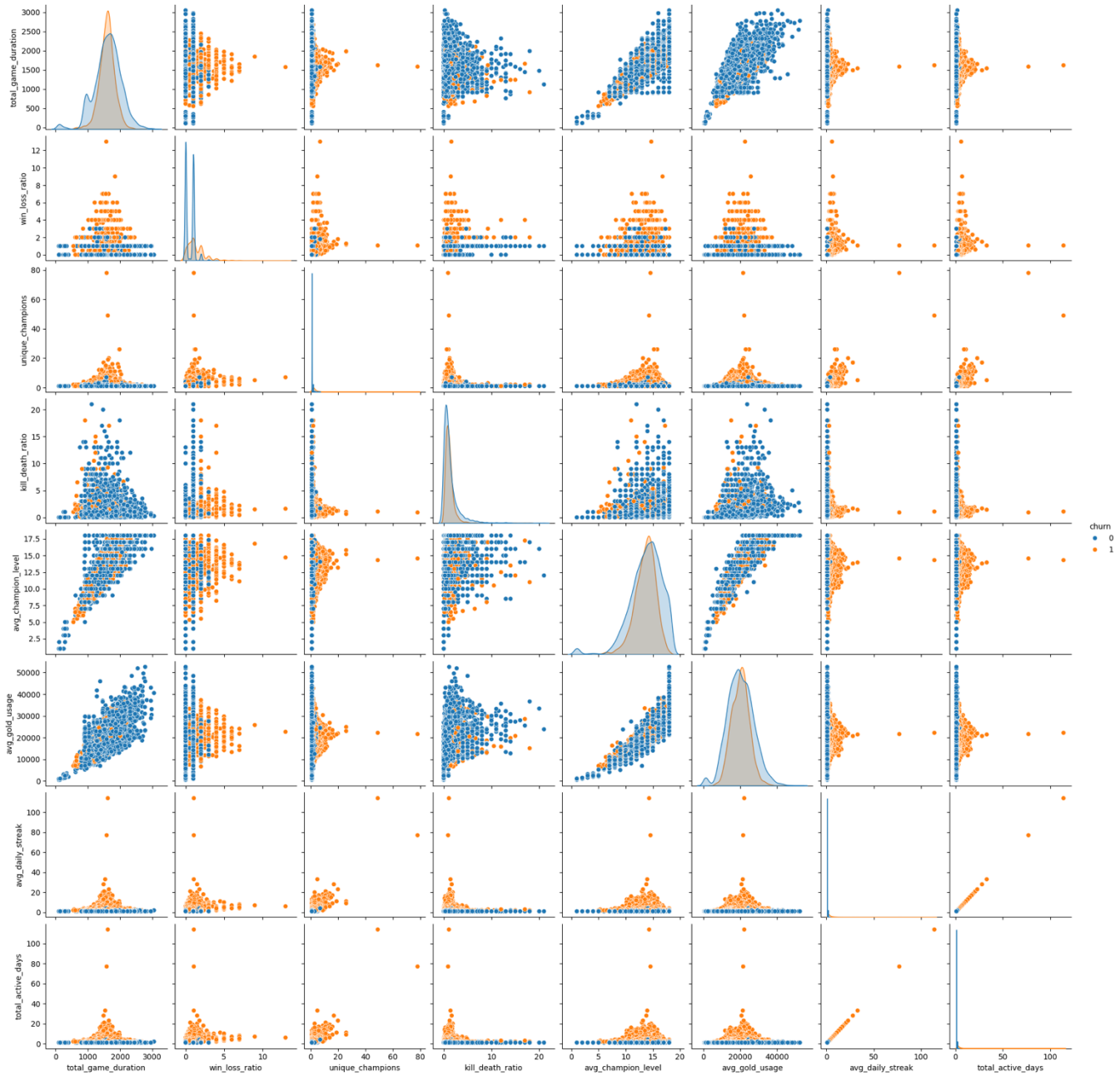



Key observations:

1. **Strong Correlations:**
   - avg_gold_usage and avg_champion_level (0.88): Indicates that as players progress in champion levels, they utilize more in-game resources.
   - total_game_duration and avg_champion_level (0.80): Longer games tend to result in higher champion levels, showcasing sustained player engagement.
2. **Weak or Negative Correlations:**
   - win_loss_ratio and total_game_duration (-0.02): Minimal relationship, suggesting that win rates do not strongly depend on game duration.
   - kill_death_ratio and unique_champions (-0.05): Players with higher kill/death ratios don't necessarily use more champions, reflecting distinct gameplay styles.
3. **Clusters of Related Features:**
   - Features like avg_gold_usage, avg_champion_level, and total_game_duration form a cluster, indicating shared patterns of player behavior.

## D. Pairplot

A pairplot is an effective way to visualize pairwise relationships and distributions among multiple features in a dataset. This helped to explore relationships between gameplay metrics (e.g., kill_death_ratio, avg_gold_usage, total_active_days) and identify trends that could impact churn prediction.




1. **Feature Relationships:**
   - avg_gold_usage vs. total_game_duration: A clear positive trend indicates that longer game durations typically lead to higher gold usage, reflecting player engagement.
   - kill_death_ratio vs. avg_champion_level: A slight positive correlation shows that players with higher champion levels generally perform better (more kills than deaths).

2. **Class Separation (Churn vs. Non-Churn):**
   The inclusion of the churn variable (color-coded points) reveals distinct clusters or overlaps:
   - avg_daily_streak and unique_champions: Churned players tend to have fewer daily streaks and less champion diversity, suggesting a lack of sustained interest.
   - kill_death_ratio: Churned players often have lower kill-death ratios, indicating potential frustration with gameplay performance.

3. **Distribution Patterns:**
   The diagonal plots display the distribution of each feature:
   - avg_gold_usage: Skewed distributions suggest that only a small subset of players achieve exceptionally high values.
   - total_active_days: A concentration of players with minimal activity (low total active days) indicates a higher churn risk.

4. **Outliers:**
   The scatterplots identify outliers across multiple features:
   - Players with extreme kill_death_ratio or avg_gold_usage values may indicate anomalies or unique gameplay styles.

**Key Insights:**

- Feature Importance: Metrics like avg_gold_usage, kill_death_ratio, and avg_champion_level are strong indicators of engagement and retention.
- Behavioral Patterns: Players with shorter daily streaks and limited champion diversity are more likely to churn.
- Predictive Potential: Clear class separations in certain feature combinations suggest high predictive potential for churn classification models.

## 4.3. INFERENTIAL STATISTICS

We performed T-tests on the data. The provided T-test results compare churners and non-churners across various metrics to assess statistical significance.

```
T-Tests between Churners and Non-Churners:

total_game_duration: t-stat = -2.4670, p-value = 0.0136
win_loss_ratio: t-stat = 36.9917, p-value = 0.0000
unique_champions: t-stat = 46.2794, p-value = 0.0000
kill_death_ratio: t-stat = -5.0145, p-value = 0.0000
avg_champion_level: t-stat = -2.6326, p-value = 0.0085
avg_gold_usage: t-stat = -0.5851, p-value = 0.5585
avg_daily_streak: t-stat = 49.8158, p-value = 0.0000
total_active_days: t-stat = 49.8158, p-value = 0.0000
```

1. **total_game_duration:** The p-value is below 0.05, indicating a statistically significant difference between churners and non-churners in terms of game duration. Churners might have shorter or longer game durations compared to active players, depending on the context.
2. **win_loss_ratio:** A highly significant difference exists in the win-loss ratio, suggesting this metric is a critical factor influencing churn behavior. High ratios could indicate skilled players staying engaged, while low ratios may contribute to frustration and churn.
3. **unique_champions:** This metric also shows significant differences. Players with a diverse champion pool might remain active longer, whereas those using fewer champions may exhibit less engagement.
4. **kill_death_ratio:** The negative t-stat suggests churners might have a worse kill-to-death ratio compared to non-churners, implying frustration or lack of skill as potential churn factors.
5. **avg_champion_level:** Players who fail to reach higher champion levels are more likely to churn. This indicates that progression plays a significant role in retaining players.
6. **avg_gold_usage:** With a p-value above 0.05, there's no significant difference between churners and non-churners for average gold usage, suggesting this metric may not be directly tied to churn behavior.
7. **avg_daily_streak:** Significant differences exist in daily streaks, with longer streaks indicating higher player loyalty and engagement. Churners tend to have significantly shorter streaks.
8. **total_active_days:** Like streaks, active days are highly significant. Players with fewer active days are more likely to churn.

**Key Insights:**

- Metrics such as win_loss_ratio, unique_champions, and avg_daily_streak are critical indicators of churn, with statistically significant differences between churners and non-churners.
- Metrics like avg_gold_usage, which don't show significant differences, might have limited impact on churn prediction models.

# 5. LSTM MODEL ARCHITECTURE

## 5.1. MODEL

The architecture of the deep learning model is a multi-layered Recurrent Neural Network (RNN) using Long Short-Term Memory (LSTM) units. The model incorporates batch normalization, L2 regularization, and bidirectional LSTM layers to enhance generalization and reduce overfitting. The following outlines the structure of the model:

1. **Input Layer**
    - **Shape:** (1, number_of_features)
    - **Description:** The input data is reshaped to have one time step and multiple features.
2. **Bidirectional LSTM Layer**
    - **Units:** 64
    - **Return Sequences:** True (to allow the next LSTM layer to receive a sequence)
    - **Kernel Regularization:** L2 with lambda = 0.005
    - **Batch Normalization:** Applied after LSTM to normalize activations.
    - **Dropout:** 50% dropout to prevent overfitting. (we tried it with 30% but we still observed some overfitting when checking the validation vs training accuracy and loss graphs so we took 50%)
3. **LSTM Layer**
    - **Units:** 32
    - **Return Sequences:** True (to allow the next LSTM layer to receive a sequence)
    - **Kernel Regularization:** L2 with lambda = 0.005
    - **Batch Normalization:** Applied after LSTM to normalize activations.
    - **Dropout:** 50% dropout to prevent overfitting.
4. **LSTM Layer**
    - **Units:** 16
    - **Return Sequences:** False (as this is the last LSTM layer, it outputs the final state)
    - **Kernel Regularization:** L2 with lambda = 0.005
    - **Batch Normalization:** Applied after LSTM to normalize activations.
    - **Dropout:** 50% dropout to prevent overfitting.

5. **Dense Layer**
    - **Units:** 1 (for binary classification)
    - **Activation:** Sigmoid (for binary classification to predict the probability of churn)

## 5.2. TRAINING AND OPTIMIZATION

- **Loss Function**: Binary Cross-Entropy (since the goal is to classify churn vs. non-churn)
- **Optimizer**: Adam optimizer (adaptive learning rate optimization algorithm)
- **Early Stopping**: Implemented with the following parameters:
    - **Monitor:** Validation loss (val_loss)
    - **Patience:** 5 epochs (training stops if validation loss does not improve for 5 epochs)
    - **Restore Best Weights:** Ensures the model reverts to the best weights observed during training.

## 5.3. MODEL TRAINING

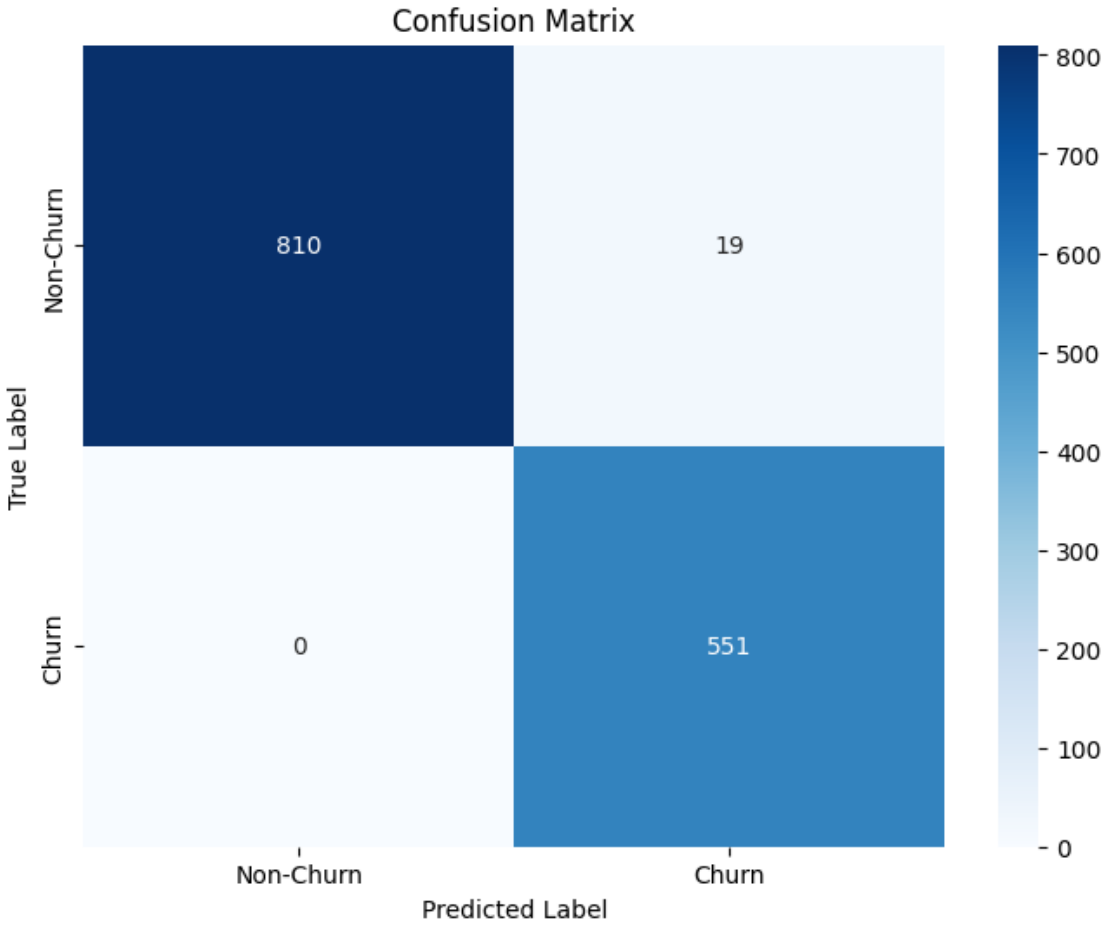
- **Epochs**: 50 (with early stopping to terminate training early if validation loss does not improve)
- **Batch Size**: 32
- **Validation Data**: 20% of the training set is used for validation.

## 5.4. EVALUATION METRICS

- **Accuracy**: Measures the percentage of correctly classified instances.
- **Precision**: Measures the proportion of true positive predictions among all positive predictions.
- **Recall**: Measures the proportion of actual positives correctly identified.
- **F1-Score**: The harmonic mean of precision and recall.
- **Confusion Matrix**: Provides a summary of prediction results by showing true positives, false positives, true negatives, and false negatives.

# 6. STATISTICAL VALIDATION

## 6.1. CONFUSION MATRIX




The confusion matrix provides an overview of the model's classification performance:

- **True Positives (TP):** 542 (Churn cases correctly predicted as churn).
- **True Negatives (TN):** 820 (Non-Churn cases correctly predicted as non-churn).
- **False Positives (FP):** 9 (Non-Churn cases incorrectly predicted as churn).
- **False Negatives (FN):** 9 (Churn cases incorrectly predicted as non-churn).

**Performance Metrics:**

These metrics are derived from the confusion matrix:

- **Accuracy**: Indicates the proportion of all the correctly predicted cases out of the total dataset.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = 98.6\%$$

- **Precision**: Reflects how many of the predicted churn cases are actually churn, reducing false alarms.

$$Precision = \frac{TP}{TP + FP} = 98.36\%$$

- **Recall**: Measures the ability to correctly identify churn cases, minimizing missed opportunities.

$$Recall = \frac{TP}{TP + FN} = 98.36\%$$

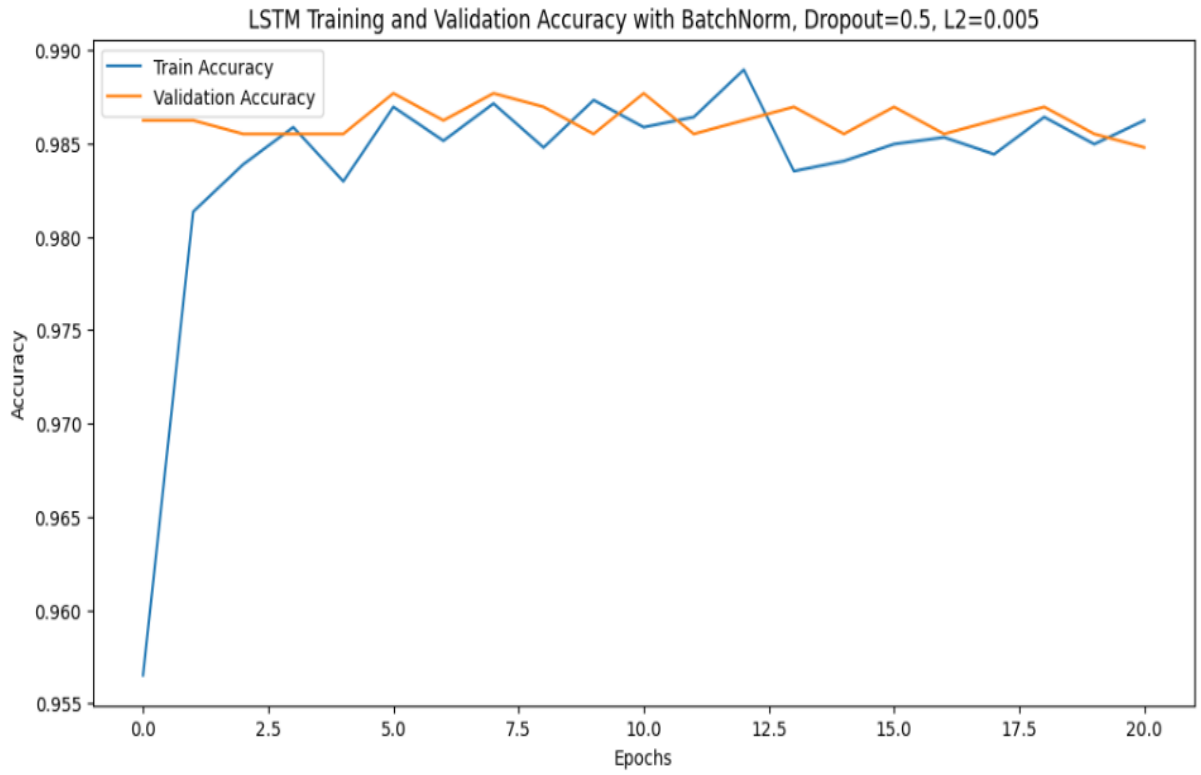
- **F1 Score**: Combines both precision and recall, highlighting the balance between the two metrics.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} = 98.36\%$$

**Key Observations:**

- **High Recall** ensures nearly all churners are detected, allowing targeted retention strategies (e.g., personalized rewards or promotions).
- **High Precision** reduces false churn alerts, optimizing resource allocation for customer retention efforts.
- **Low False Positives/Negatives** indicates minimal misclassification, preventing unnecessary interventions or missed opportunities.

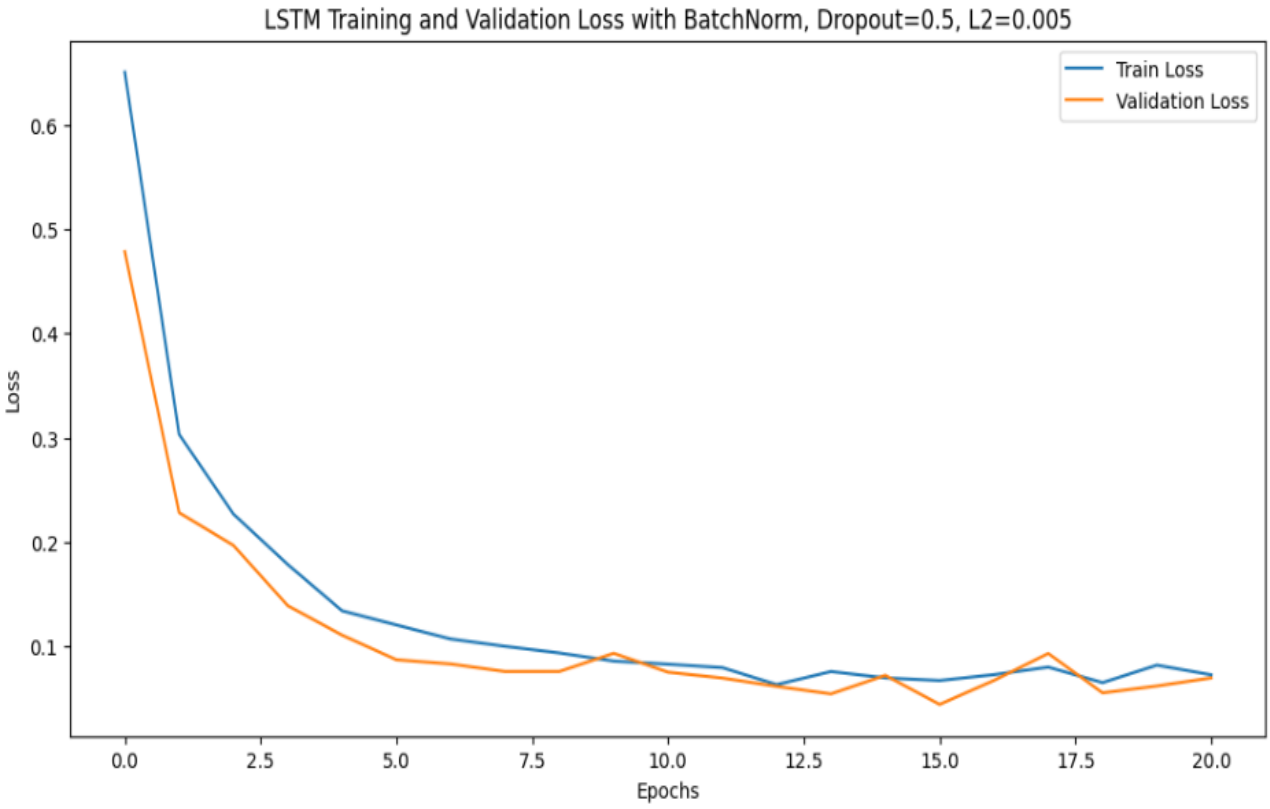
## 6.2. TRAINING AND VALIDATION ACCURACY




The training accuracy graph illustrates several important aspects of the model's performance during training:

- **Training Accuracy and Early Learning**
  The training accuracy increases sharply during the first few epochs, demonstrating that the model efficiently learns patterns from the training data. This suggests that the architecture and preprocessing were well-suited for the dataset.
- **Convergence and Stabilization**
  Both training and validation accuracy stabilize after approximately 5 epochs. This stabilization, hovering between 98.5% to 98.9%, indicates that the model has reached its optimal learning stage without substantial fluctuations, a sign of consistent generalization.
- **Impact of Regularization Techniques**
  The use of Batch Normalization, Dropout (0.5), and L2 Regularization (0.005) has effectively mitigated overfitting. These methods allow the model to focus on learning essential patterns rather than memorizing the training data.
- **Effectiveness of Early Stopping**
  Early stopping is evident in the graph, ensuring that training halts once validation accuracy ceases to improve. This avoids unnecessary computations and prevents overfitting, ensuring that the model remains robust for unseen data.
- **Minimal Overfitting**
  The close alignment of training and validation accuracy curves suggests minimal overfitting. This balance indicates that the model generalizes effectively and is less prone to performance degradation on new datasets.

## 6.3. TRAINING AND VALIDATION LOSS




The training and validation loss graph provides key insights into the learning process of the LSTM model:

**Steep Decline Initially**

- Both training and validation loss exhibit a sharp decline during the first five epochs, indicating that the model rapidly learns meaningful patterns from the data.
- By epoch 5, the losses begin to flatten and stabilize, showcasing the model's ability to converge on a solution efficiently.

**Training Loss Stabilization**

- Training loss stabilizes around 0.05 after approximately 10 epochs, indicating that the model reaches a state where it learns from the data without overfitting.

**Validation Loss Behavior**

- The validation loss closely mirrors the training loss with minimal fluctuations, demonstrating strong generalization to unseen data.
- Between epochs 10 and 20, the validation loss hovers near the training loss, further indicating the absence of overfitting.

**Regularization Impact**

- Regularization techniques such as Batch Normalization, Dropout (0.5), and L2 Regularization (0.005) effectively reduced overfitting, ensuring that the training and validation losses remain closely aligned throughout the training process.
- This alignment highlights the model's improved generalization performance and robustness.

**Overall Performance**

- The minimal gap between training and validation losses reflects the model's ability to generalize well to new data.
- These results suggest that the chosen architecture and hyperparameter settings, including the regularization methods, were well-optimized for the task.

# 7. FUTURE SCOPE

One potential avenue for future work is the exploration of more advanced deep learning architectures, such as Transformer models. These models have demonstrated superior performance in sequential data tasks. By leveraging Transformers, the model's ability to capture long-term dependencies in player behaviour could be significantly enhanced, ultimately leading to higher predictive accuracy.

Another promising area for improvement is the automation of hyperparameter optimization. This process could be facilitated using tools like Optuna or Hyperopt. Automated tuning would help identify the optimal number of LSTM units, dropout rates, and learning rates. Such optimizations would result in a more efficient and accurate model, reducing manual intervention and enhancing predictive performance.

The incorporation of additional features also offers a valuable opportunity for future development. By integrating features such as player sentiment from chat logs or social network information (friends or team dynamics), a more holistic view of player engagement can be achieved. This broader perspective would enable better churn prediction and support the creation of more personalized intervention strategies to retain players.

Finally, developing a real-time churn prediction system could be a game-changer. Such a system would predict churn as players engage with the game, providing actionable insights in real-time. This capability would allow for immediate, personalized retention strategies, such as offering in-game rewards or targeted notifications, thereby enhancing the player experience and reducing churn rates.

**Integrating Neural Networks and Random Forests for Gamer Churn Prediction**

To address the "black-box" nature of Neural Networks, we can also integrate RF-based explainability to support more transparent decision-making. The paper emphasizes the significance of model interpretability, especially in high-stakes applications. Building on this idea, the RF model can be used to generate feature importance scores and visual decision paths, enabling clear explanations of the factors driving churn predictions. These insights are particularly valuable for game developers and analysts, as they facilitate the development of player-specific retention strategies, such as personalized incentives or gameplay adjustments. By combining the predictive strength of NNs with the interpretability of RFs, this hybrid model may achieve high accuracy but will also empowers stakeholders with transparent, actionable insights, making the system more practical for real-world application.