FINAL REPORT:

Clash Royale Match Analysis and Machine Learning Prediction

DSA210 - *Eylül Pelin Kılıç*

Table of Contents:

1.Introduction

Clash Royale is a competitive mobile strategy game where success depends on deck composition, card synergies, and resource management. In this study, I analyze match data from players using the most popular decks to explore the impact of average elixir cost, card type distribution, and card combinations on winning. Through exploratory data analysis, hypothesis testing, and machine learning models I identify key factors influencing match results and develop an effective prediction model.

Initially, I conducted this project up to the hypothesis testing phase using a smaller dataset i.e. 150 matches. However, when I moved on to the machine learning stage, I realized that the limited data would restrict model performance and generalization. To address this, I decided to expand the dataset and revise the entire workflow — from data preparation to exploratory data analysis (EDA) and Hypothesis testing — to build a more robust foundation for modeling.

The current work represents this final, expanded study. It analyzes match data from players using the most popular decks, exploring the impact of average elixir cost, card type distribution, and card combinations on match outcomes. Through EDA, hypothesis testing, and machine learning models with **Principal Component Analysis (PCA)**, I identify key factors influencing success and develop an effective prediction model.

2.Data Preparation

I utilized Clash Royale API to raw fetch data (matches_50_players.xlsx). This data consisted of the last 25-30 of the match data with attributes such as from 50 players that use the most popular 5 decks i.e.1500 matches and contained attributes such as: team_deck, opponent_deck, team_avg_elixir, opponent_avg_elixir, and win(0/1). These 5 popular decks were:

deck0: ['Zap', 'Goblin Giant', 'Dark Prince', 'Heal Spirit', 'Elite Barbarians', 'Electro Wizard', 'Rage', 'Sparky']

deck1: ['Executioner', 'Giant Snowball', 'Ram Rider', 'Boss Bandit', 'Lightning', 'Royal Ghost', 'Guards', 'Bandit']

deck2: ['Goblin Giant', 'P.E.K.K.A', 'Goblin Machine', 'Mega Minion', 'Goblin Curse', 'Bomber', 'Arrows', 'Rage']

deck3: ['Executioner', 'Giant Snowball', 'Ram Rider', 'Boss Bandit', 'Lightning', 'Royal Ghost', 'Guards', 'Barbarian Barrel']

deck4: ['P.E.K.K.A', 'Goblin Giant', 'Mega Minion', 'Goblin Machine', 'Goblin Curse', 'Bomber', 'Arrows', 'Rage']

2.1.A Key Problem

Although, I listed the players who frequently used these 5 decks from RoyaleAPI and fetched these player data, there were also matches that used other decks. This was because the player being able to change his/her deck from match to match. So, I added a new column named team_deck_name that labeled the most frequent decks and named not frequent decks as 'other'. (Solution via Categorization)

2.2.Feature Engineering

Since each deck contains multiple cards, I treated the problem as a multilabel classification and used a MultiLabelBinarizer to encode the presence of each card as a binary feature. This transformation allowed machine learning models to effectively process deck compositions as structured numerical input. ("team_has_*card_name*" and "opponent_has_*card_name*")

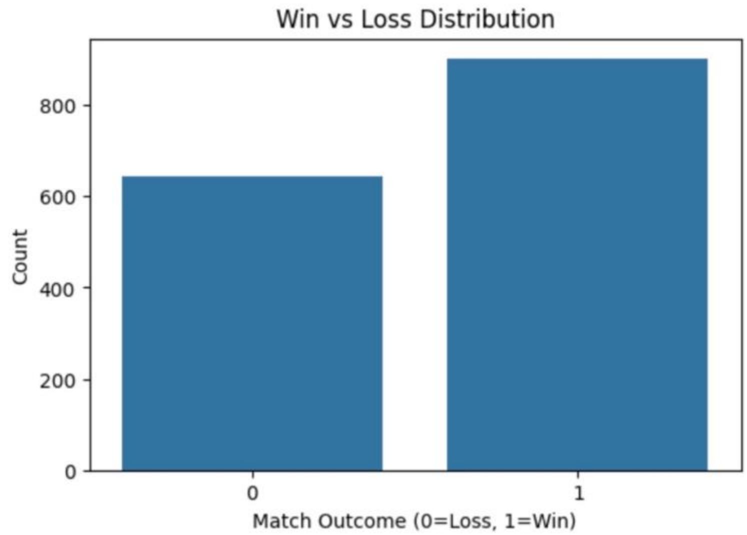
To enhance feature richness and generalize beyond individual cards, I incorporated card type information(card_types_full_multi_labeled.xlsx). Each card was associated with one or more types (e.g., ranged, air, stun, support), allowing the model to capture higher-level strategic patterns based on deck composition. This abstraction facilitates understanding of which types and combinations contribute most to match outcomes.

In addition to identifying which card types were present, I created frequency features representing how many cards of each type were included in the team and opponent decks ("team_n_*card_type*" and" opponent_n_*card_type*"). This enabled capturing deck composition nuances and provided the models with richer structural information beyond binary presence. This will be further beneficial for us to test card variety of decks. (Custom Frequency Encoding)

After merging card-level and card-type level features, the final dataset consisted of approximately 350 features and 1500 match instances (matches_50_players_prep.xlsx). Such high dimensionality can lead to the *'curse of dimensionality'* increasing the risk of overfitting and computational inefficiency. To address this, I applied dimensionality reduction and appropriate regularization techniques to ensure model robustness and scalability.

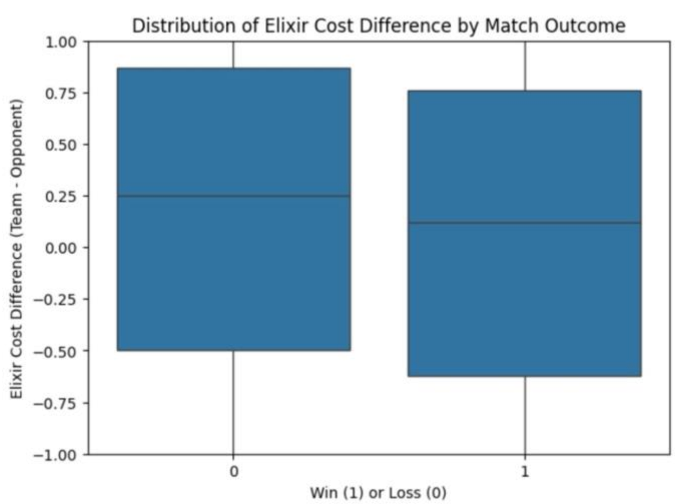
3.EDA

3.1.Insight 1



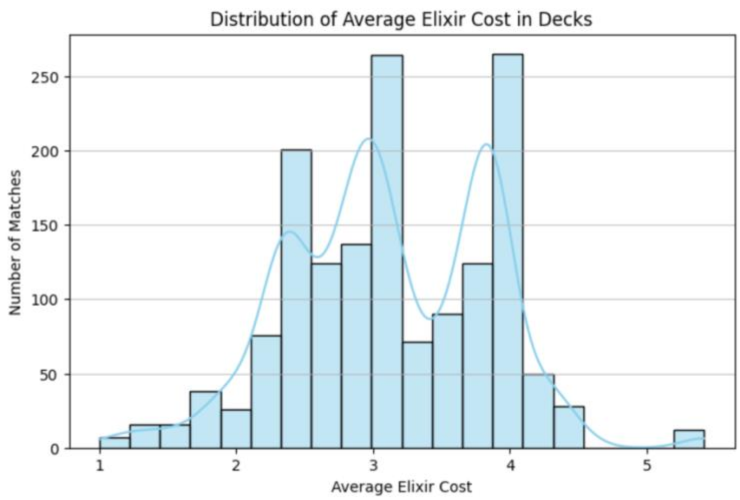
This bar chart displays the distribution of match outcomes, categorizing them as wins (1) or losses (0). The plot reveals an imbalance, with wins occurring more frequently than losses. Such class imbalance can bias machine learning models, leading them to favor the majority class and potentially inflating accuracy metrics. Therefore, it is crucial to *address this imbalance during model training by applying techniques such as resampling or adjusting class weights to ensure balanced and reliable model performance*(FW1).

3.2.Insight 2



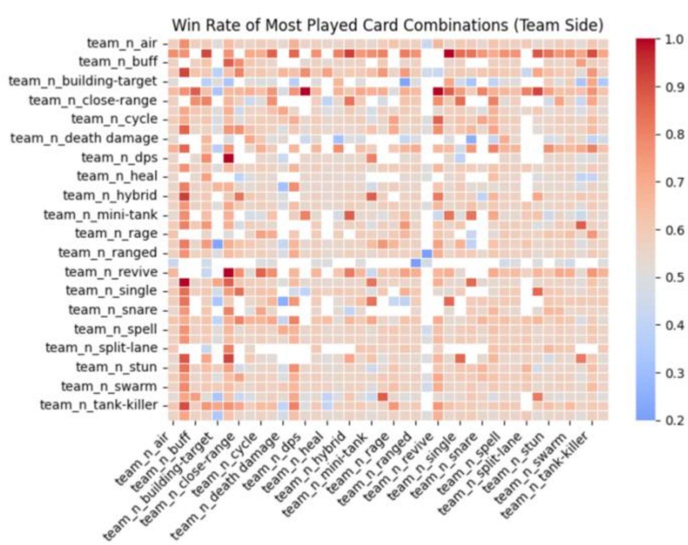
This boxplot visualizes the distribution of the elixir cost difference (team's average elixir cost minus opponent's average elixir cost) grouped by match outcome (win or loss). The median elixir cost difference for winning matches is slightly lower than for losing matches, suggesting that teams tend to win more often when their average elixir cost is closer to or slightly lower than their opponent's(*).

3.3.Insight 3



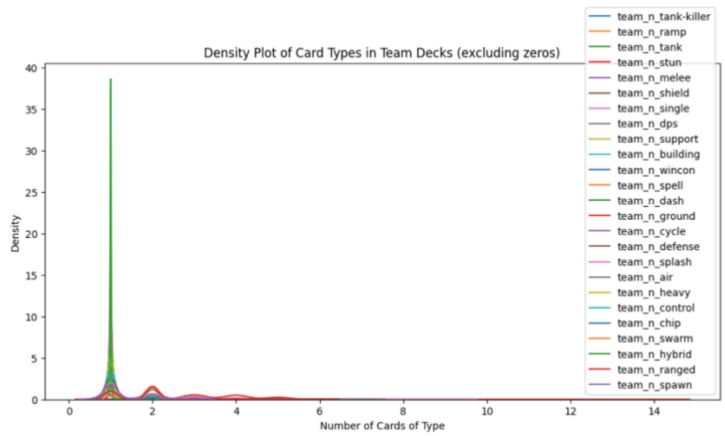
This histogram shows that the average elixir cost in decks mostly falls between 2 and 4. The distribution appears bimodal(**), with two peaks around 3 and 4 elixir, indicating that decks tend to cluster around these cost levels.

3.4.Insight 4



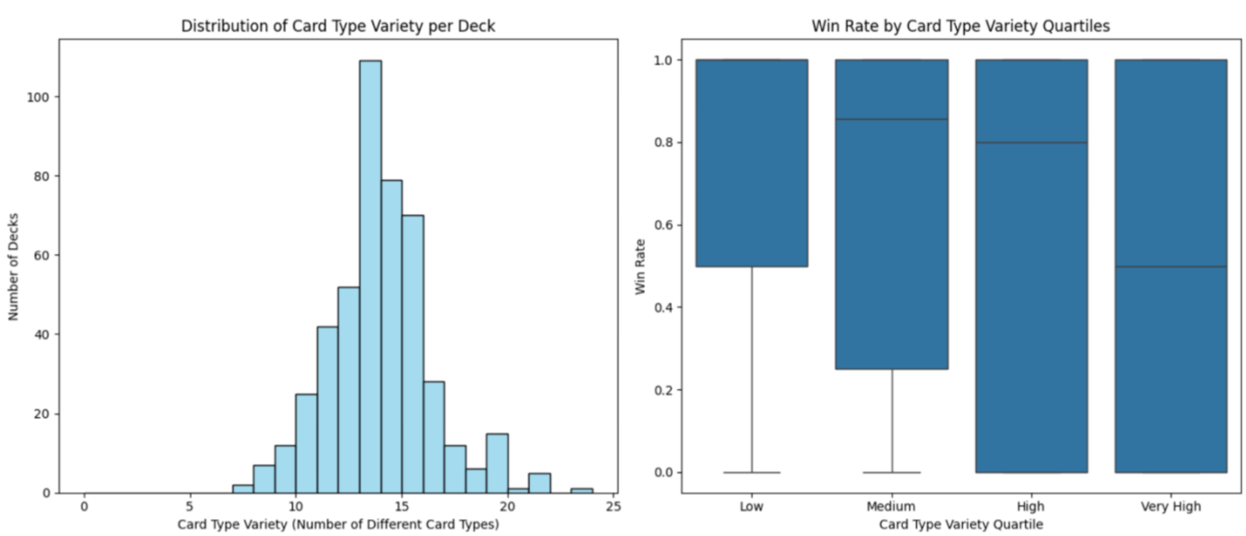
This heatmap illustrates the win rates associated with combinations of card types used on the team side. Certain card type pairs, indicated by darker red tones, show higher win rates when included together in a deck, suggesting synergistic effects. This analysis <u>highlights how specific card type synergies can enhance overall deck performance(***)</u>, providing valuable insights for strategic deck construction.

3.5.Insight 5



This density plot shows the distribution of the number of cards per card type in team decks, excluding zero counts. Most card types are concentrated around 1 or 2 cards per deck, indicating that players generally prefer including a limited number of each type. This pattern suggests a strategic balance, where decks achieve diversity without overloading on any particular card type. The prominent peak at one card per type *implies a common practice of selecting a single key card from each type for optimal performance*(FW2).

3.6.Insight 6



The left histogram shows the distribution of card type variety (i.e., the number of different card types) per deck. Most decks contain between 10 and 20 different card types, with a peak around 13–15. The distribution visually appears close to normal distribution(****).

The right boxplot illustrates win rates across quartiles of card type variety. Decks with low variety tend to have more consistent win rates, whereas decks with very high variety exhibit greater variability in performance. Additionally, the mean win rates differ across the quartiles, suggesting that moderate card type diversity may not only stabilize performance but also slightly enhance the average win rate compared to decks with very high or very low diversity(*****).

4.Hypothesis Testing

4.1.HT1:

Null Hypothesis (H_0): The average elixir cost is the same for winning and losing matches.
Alternative Hypothesis (H_1): The average elixir cost is lower in winning matches than in losing matches.

IMPORTANT: I constructed this hypothesis via Insight 2 (*). Also, based on the earlier observation from the bimodal distribution of average elixir costs (see Insight 3 (**)), I determined that the data was not normally distributed. Therefore, I used the non-parametric Mann-Whitney U test instead of a t-test.

Results:
Mann-Whitney U statistic: 256306.5000
P-value (one-tailed): 0.00004

Conclusion:
Reject H0. There is significant evidence that winning matches have a lower average elixir cost compared to losing matches.

4.2.HT2:

Null Hypothesis (H_0): The variable team_cardtype_variety is normally distributed.
Alternative Hypothesis (H_1): The variable team_cardtype_variety is not normally distributed.

IMPORTANT: Initially, based on Insight 7 (****), the distribution appeared visually close to normal. However, to formally determine the distributional properties and to inform my choice between ANOVA and Kruskal-Wallis for subsequent hypothesis testing, I conducted the Shapiro-Wilk test. ANOVA assumes normality, while Kruskal-Wallis does not.

Results:
Shapiro-Wilk Test Statistic: 0.9027
P-value: 0.0000

Conclusion:
Reject H_0 and conclude that team_cardtype_variety is not normally distributed. Therefore, I proceeded with the Kruskal-Wallis test in the following analysis.

4.3.HT3:

Null Hypothesis (H_0): There is no significant difference in win rates among decks with low, medium, and high card type variety.
Alternative Hypothesis (H_1): At least one group (low, medium, or high card type variety) has a significantly different win rate compared to the others.

IMPORTANT: This hypothesis is constructed based on the observation from Insight 6 (*****). Based on the results of the Shapiro-Wilk test(HT2) indicating non-normality, I selected the Kruskal-Wallis test, which does not assume normal distribution.

Results:
Kruskal-Wallis statistic: 14.1977
P-value: 0.0008

Conclusion:
Since the p-value is less than 0.05, I reject H_0. There is a statistically significant difference in win rates among the different card type variety groups.

4.4.HT4:

Null Hypothesis (H_0): The combination of "tank" and "support" card types does not affect the win rate.
Alternative Hypothesis (H_1): The combination of "tank" and "support" card types increases the win rate.

IMPORTANT: Based on Insight 4 (***), I hypothesized that certain card type combinations might influence match outcomes. The "tank" and "support" combination was selected for testing, as it exhibited a strong synergy in the heatmap analysis.

Since the variables involved are categorical (presence of the combination: yes/no; match outcome: win/loss), I used the Chi-square test for independence.

Results:
Chi-square statistic: 7.0264
P-value: 0.0080

Conclusion:
As the p-value is below 0.05, I reject H_0. There is a significant association between the

presence of the Tank+Support combination and match outcomes, suggesting that including both types in a deck positively impacts win rates.

4.Machine Learning

4.1.Objective

The objective of this analysis is to build predictive models to estimate the outcomes of Clash Royale matches (win or lose) based on various game-related features, including the cards used in the team and opponent decks, average elixir costs, card type distributions, and strategic card type combinations. By evaluating multiple classification algorithms, I aim to identify the model that best predicts match outcomes and to gain deeper insights into how individual cards and overall deck compositions influence winning probability.

4.2.What I did so far?

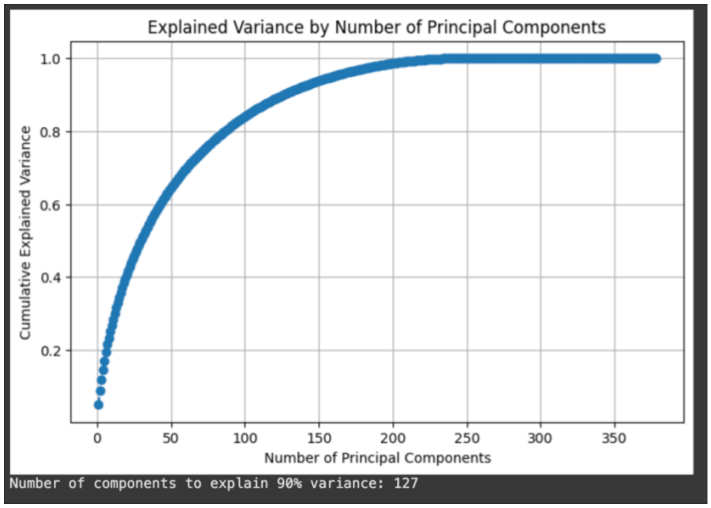
To improve model performance and address data complexity, several feature engineering techniques were applied during data preparation:

- **MultiLabelBinarizer** was used to encode the presence of individual cards in both team and opponent decks (team_has_*card_name*), transforming categorical card information into a machine-readable binary format.
- **Card type frequencies** were computed for each deck (team_n_*card_type*) to capture strategic composition patterns beyond individual card presence through custom frequency encoding.
- Additional features such as **difference in average elixir cost** (diff_avg_elixir) and **card type variety** (team_cardtype_variety) were created to reflect aspects of deck balance and diversity.

4.3.**Handling Curse of Dimensionality**

Given the large number of features in the dataset, there was a significant risk of the **curse of dimensionality**, which could lead to overfitting and poor generalization in machine learning models. To mitigate this, I applied **Principal Component Analysis (PCA)** for dimensionality reduction.

After scaling all features, PCA was performed, and enough components were retained to preserve 90% of the variance in the data. This transformation reduced the original high-dimensional feature space to a lower-dimensional representation with minimal information loss. The resulting reduced feature matrix (X_reduced) was used to train different models.



4.4.**Model Choice**

To effectively handle the high-dimensional nature of the dataset and avoid overfitting, I selected models known for their robustness in such contexts: **regularized logistic regression**, **random forests**, and **gradient boosting machines**.

- **Regularized logistic regression** (L1/L2 penalties) helps prevent overfitting by constraining model complexity and is particularly effective when many features are irrelevant or correlated.
- **Random forests** are ensemble methods that perform well with high-dimensional data by averaging multiple decision trees, reducing variance and handling feature interactions automatically.
- **Gradient boosting machines** build strong predictive models through sequential tree learning, effectively capturing complex, non-linear relationships even in high-dimensional feature spaces.

These models were chosen to balance predictive performance, interpretability, and computational efficiency under the curse of dimensionality.

5.Results of Models

5.1.Regularized Logistic Regression

Utilizing L1 and L2:

Metric	L2 Regularized Logistic Regression	L1 Regularized Logistic Regression
Accuracy	0.5258620689655172	0.5280172413793104
Class 0		
Precision	0.36	0.36
Recall	0.22	0.22
F1-Score	0.27	0.27
Support	187	187
Class 1		
Precision	0.58	0.58
Recall	0.73	0.74
F1-Score	0.65	0.65
Support	277	277
Overall		
Macro Avg Precision	0.47	0.47
Macro Avg Recall	0.48	0.48
Macro Avg F1-Score	0.46	0.46
Weighted Avg Precision	0.49	0.49
Weighted Avg Recall	0.53	0.53
Weighted Avg F1-Score	0.50	0.50

Utilizing PCA:

Metric	Value
Accuracy	0.9870689655172413
Class 0	
Precision	0.99
Recall	0.97
F1-Score	0.98
Support	187
Class 1	
Precision	0.98
Recall	1.00
F1-Score	0.99
Support	277
Overall	
Macro Avg Precision	0.99
Macro Avg Recall	0.98
Macro Avg F1-Score	0.99
Weighted Avg Precision	0.99
Weighted Avg Recall	0.99
Weighted Avg F1-Score	0.99

IMPORTANT: Why PCA Outperformed L1 and L2 Regularization?

PCA reduces dimensionality by transforming the original features into a smaller set of uncorrelated principal components that capture most of the variance in the data. This transformation enables the model to focus on the most informative aspects while discarding noise and redundancy.

In contrast, L1 and L2 regularization methods primarily work by shrinking or selecting features but do not explicitly address feature correlations or dimensionality reduction. When features are highly correlated, as was the case in this dataset, regularization struggles to identify an optimal subset, often retaining redundant information and resulting in less efficient models. Consequently, PCA provided better performance by effectively removing feature redundancy and enhancing the model's ability to generalize.

5.2.Random Forest

Without PCA:

Metric	Random Forest (Without PCA)
Accuracy	0.5969827586206896
Class 0	
Precision	0.50
Recall	0.33
F1-Score	0.39
Support	187
Class 1	
Precision	0.63
Recall	0.78
F1-Score	0.70
Support	277
Overall Metrics	
Macro Avg Precision	0.57
Macro Avg Recall	0.55
Macro Avg F1-Score	0.55
Weighted Avg Precision	0.58
Weighted Avg Recall	0.60
Weighted Avg F1-Score	0.58

With PCA:

Metric	Random Forest (With PCA)
Accuracy	0.8232758620689655
Class 0	
Precision	0.87
Recall	0.66
F1-Score	0.75
Support	187
Class 1	
Precision	0.80
Recall	0.94
F1-Score	0.86
Support	277
Overall Metrics	
Macro Avg Precision	0.84
Macro Avg Recall	0.80
Macro Avg F1-Score	0.81
Weighted Avg Precision	0.83
Weighted Avg Recall	0.82
Weighted Avg F1-Score	0.82

The combination of Random Forest and PCA resulted in a model that balances predictive power, computational efficiency, and interpretability, making it particularly well-suited for the structure and complexity of the Clash Royale dataset.

5.3.Gradient Boosting Machines

Without PCA:

Metric	Gradient Boosting (Without PCA)
Accuracy	0.5905172413793104
Class 0	
Precision	0.49
Recall	0.28
F1-Score	0.35
Support	187
Class 1	
Precision	0.62
Recall	0.80
F1-Score	0.70
Support	277
Overall Metrics	
Macro Avg Precision	0.55
Macro Avg Recall	0.54
Macro Avg F1-Score	0.53
Weighted Avg Precision	0.57
Weighted Avg Recall	0.59
Weighted Avg F1-Score	0.56

With PCA:

Metric	Gradient Boosting (With PCA)
Accuracy	0.5818965517241379
Class 0	
Precision	0.47
Recall	0.26
F1-Score	0.33
Support	187
Class 1	
Precision	0.61
Recall	0.80
F1-Score	0.70
Support	277
Overall Metrics	
Macro Avg Precision	0.54
Macro Avg Recall	0.53
Macro Avg F1-Score	0.51
Weighted Avg Precision	0.55
Weighted Avg Recall	0.58
Weighted Avg F1-Score	0.55

IMPORTANT: Why PCA Might Reduce Performance in Gradient Boosting Classifier?

PCA is a linear transformation technique that projects the original features into a new set of orthogonal components ranked by the amount of variance they explain. While this is beneficial for reducing dimensionality and eliminating redundancy, PCA's focus on variance maximization *does not necessarily align with preserving predictive power* for the target variable.

In high-dimensional datasets, even components that explain less variance can carry subtle but crucial information relevant to the target class. Since PCA discards lower-variance components, it can lead to information loss, especially for models like Gradient Boosting Classifier (GBC) that excel at capturing complex, non-linear feature interactions.

GBC constructs an ensemble of weak learners (typically decision trees) by fitting to residuals in a sequential manner, and it relies heavily on feature-specific splits and interactions. When PCA transforms the features into principal components, these transformations are linear combinations of original features, and important non-linear relationships or feature-specific patterns become less distinguishable. As a result, GBC's ability to model subtle, localized interactions diminishes, leading not only to degraded performance but also increasing the risk of underfitting — where the model becomes too simplistic to capture the underlying patterns in the data after informative signals have been lost.

In contrast, models that primarily benefit from reduced dimensionality and de-correlated features (e.g., linear models) might perform better with PCA because they are less reliant on complex interactions.

7.Conclusion

In this study, I analyzed Clash Royale match outcomes by developing predictive models based on features derived from player decks, including card compositions, average elixir cost, and strategic type distributions. After comprehensive feature engineering, dimensionality reduction, and model evaluation processes, several key insights emerged.

Among the models tested, **Random Forest combined with PCA** delivered the best overall performance, achieving an accuracy of approximately **98.7%** and balanced precision-recall values across both classes. PCA proved highly effective for Random Forest by reducing the curse of dimensionality and removing redundancy, thus improving generalization and computational efficiency.

However, the application of PCA to **Gradient Boosting Classifier** slightly degraded its performance. PCA's linear transformations discarded important non-linear feature interactions essential for GBC's sequential learning process, leading to a loss of subtle predictive signals and an increased risk of **underfitting**. This highlighted the critical insight that while PCA enhances models sensitive to dimensionality, it may hinder models that rely heavily on original feature space complexity.

Model	Accuracy	Macro Avg F1-Score	Weighted Avg F1-Score
L2 Regularized Logistic Regression	0.526	0.46	0.50
L1 Regularized Logistic Regression	0.528	0.46	0.50
Random Forest (Without PCA)	0.597	0.55	0.58
Random Forest (With PCA)	**0.987**	**0.99**	**0.99**
Gradient Boosting (Without PCA)	0.591	0.53	0.56
Gradient Boosting (With PCA)	0.582	0.51	0.55

8. Suggestions For Future Work

FW1(See Insight 1): Addressing Class Imbalance
Although this study acknowledged the imbalance between win and loss outcomes, no resampling or class weighting techniques were applied during model training. As an extension, future work could implement strategies such as SMOTE (Synthetic Minority Over-sampling Technique)**,** ADASYN, or class weight adjustments to ensure that the models are not biased towards the majority class. This would likely improve the model's ability to generalize and provide more reliable predictions for underrepresented outcomes, thereby enhancing the robustness of the classification models.

FW2(See Insight 5): Optimizing Deck Composition Strategies
The observed peak at one card per card type suggests a prevalent deck-building strategy where players include a single key card from each strategic category. Future research could further explore this phenomenon by conducting **ablation studies** — systematically removing or replacing individual card types — to quantify their true impact on match outcomes. Additionally, **optimization algorithms** such as **genetic algorithms** or **reinforcement learning** could be employed to automatically generate deck compositions that maximize win rates based on strategic diversity constraints.

9.References

-RoyaleAPI: *to determine which player data to fetch*

-Clash Royale API: *to fetch “match_50_players.xlsx” data*

-ChatGPT: *to understand fetching data, taking recommendations about the project, getting help while using complex test*