CS 4641 Final Report

Team 5

Bryson Bien, Noah Brown, Marcel Dunat, Harrison Speed, Patrisiya Rumyanteseva
Github Repository: https://github.com/brysonbien/Chess-Opening-ML

1. Introduction/Background

Dataset

Chess has long been a domain where strategy, skill, and decision-making converge, and openings play a critical role in setting the foundation for the rest of the game. Chess openings, defined by the initial moves players choose, have been meticulously studied, with each opening carrying distinct advantages and risks. This project aims to build upon previous research, which has demonstrated that specific openings can impact win rates. However, the extent of this influence, especially in relation to other factors such as player skill and color choice, remains debated. Our dataset, consisting of over 20,000 games, includes features like player performance ratings, colors (white or black), opening names and ECO codes, and game results, making it ideal for exploring the influence of early-game strategies on outcomes.

Old Dataset: Chess Opening Dataset
New Dataset: Chess Dataset

Several studies have applied machine learning to analyze chess outcomes. For instance, I. Cheng developed a model that predicted game results based on openings and rating differences between players [1]. K. Raghav and L. Ahuja successfully classified games based on opening patterns and player ratings [2]. Machine learning in chess has seen remarkable milestones, including Google DeepMind's AlphaZero, a reinforcement-learning-based model that defeated the Stockfish engine within hours of training [3]. This project aims to further such research by using machine learning to explore the predictive power of openings alongside other player characteristics, potentially guiding both players and computer chess agents on how certain early-game decisions may shape the game's trajectory.

Why We Changed the Dataset from Proposal

In the original dataset, each datapoint represented a summary of data from many games that followed a specific opening strategy. Therefore, several of the features were aggregated across many games: the dataset included features like ‘player_avg’, which depended taking averages of many games’ players’ performance ratings, and ‘win_pct’, which summarized the percentage of games won as opposed to providing binary win/loss information. This made the model less interpretable and precise, and it aligned less with the goal of classifying games as wins or losses

In the new dataset, every feature represents an individual game. Predictions produced by models trained on this data will apply better to predicting the outcomes of specific games. Furthermore, this dataset is much larger, with 20,058 data points as opposed to the original 1,884. This could allow our Machine Learning models to find more complex patterns in our data and produce results with higher accuracy.

2. Problem Definition

This project focuses on binary classification of chess games as won by the black side or won by the white side based on several factors, with particular emphasis on the chosen opening strategy. The hypothesis is that certain openings, when combined with specific player ratings and colors, may significantly affect the odds of winning, allowing us to predict game results with measurable accuracy.

Motivation

By analyzing player-specific features like elo rating, game-specific attributes such as color choice, and strategic choices represented by opening moves, the project seeks to establish the role these elements play in determining game outcomes as this can help players improve their decision-making process during the planning and early stages of the game depending on various circumstances.

Plan

Visualization and Dimensionality Reduction

Initial data exploration will include visualizations to observe trends and relationships in the dataset. For example, correlation matrices can help determine which features are redundant. Manual dimensionality reduction will be used to remove redundant or irrelevant features, and dimensionality reduction algorithms, like Principal Component Analysis (PCA), will help identify and retain the most influential features to optimize the dataset for efficient analysis.

Prediction Based on Features

Opening Move Names and Other Features: By using models like a Random Forest Classifier and XGBoost, we can make predictions about the binary outcomes of chess games. This will help evaluate how well opening names and related attributes contribute to predicting game outcomes.

Further, Grid Search Cross Validation can be used to find the best hyperparameters for the Random Forest Classifier and XGBoost, which might produce predictions with higher accuracy.

Dataset Reduced with PCA: Reduced features from PCA will be tested in the predictive model to assess if condensed data retains sufficient information for accurate predictions.

GMM: Using GMM to cluster chess games could produce labels that might improve model accuracy when added as features to the dataset.

3. Methods

Preprocessing

In order to process our data so it can be used to train our machine learning models, we started by performing manual dimensionality reduction to remove features that were irrelevant or not accessible before a chess game is over. This included features like player ids or number of moves in the game. We modified the ‘opening_name’ feature to include only the primary opening strategy. We also used feature engineering to create 10 new features to store the first 5 opening moves of either side, unless an opening had fewer moves. We then tried creating a feature to represent the difference between white and black performance ratings. Next, we used a labelencoder to encode several non-numerical features. One-hot encoding was not feasible since these features contained many unique values. Additionally, we planned on using a random forest classifier, which would treat these labels as separate categories. After this initial processing, we used a correlation matrix to determine whether any features were highly correlated, which could indicate that they were redundant and some could be removed. The feature ‘moves’ was removed because it was highly correlated with ‘opening_name’ and included information that could only be obtained after a game was over.

Some additional preprocessing approaches were tested, but none of them seemed to show significant improvements in evaluation metrics. We tried several approaches for incorporating averages of different openings’ performance across the dataset. First we grouped data points by their opening names and calculated the mean of games won by the black side (this was performed on the training set and these averages were applied to the test set to avoid data leakage). Next we tried using a second dataset that contained information about average performances of different opening moves. Neither of these two approaches were included in our final report notebook, but were still documented in the file testing.ipynb.

After performing these steps, we standardized continuous columns, like the ratings and rating comparisons of the players (which is necessary for models like GMM). We also performed PCA on the dataset to reduce it to three principal components which were stored in an alternate dataframe. This would allow us to plot the components and determine if there were any obvious patterns in the dataset.

Models

Random Forest

The machine learning algorithm used was a Random Forest Classifier. We opted for a supervised learning method to make binary classifications for predicting which side will win a chess game. While a decision tree was considered, we chose an ensemble method for its higher accuracy. Random forests are ideal for detecting complex, non-linear patterns in data and providing feature importance, which helps understand the influence of features like opening moves in predicting outcomes.

Before fitting the models, K-Fold cross-validation ensured generalization to new testing data. Additionally, Grid Search cross-validation optimized hyperparameters. Three Random Forest classifiers were trained: one on manually reduced data, one on PCA-reduced data, and a third with the manually reduced dataset using the best hyperparameters from Grid Search.

Gaussian Mixture Model (GMM)

The Gaussian Mixture Model (GMM) is an unsupervised learning algorithm used to cluster data by modeling it as a mixture of Gaussian distributions. We implemented GMM to generate cluster labels as additional features for supervised learning models, enhancing classification accuracy.

Before applying GMM, we removed the target label (game outcome) to prevent data leakage. These probabilistic cluster labels were normalized alongside the original features and evaluated for their impact on predictive performance. This demonstrated how unsupervised learning techniques like GMM can uncover hidden patterns and improve overall classification accuracy.

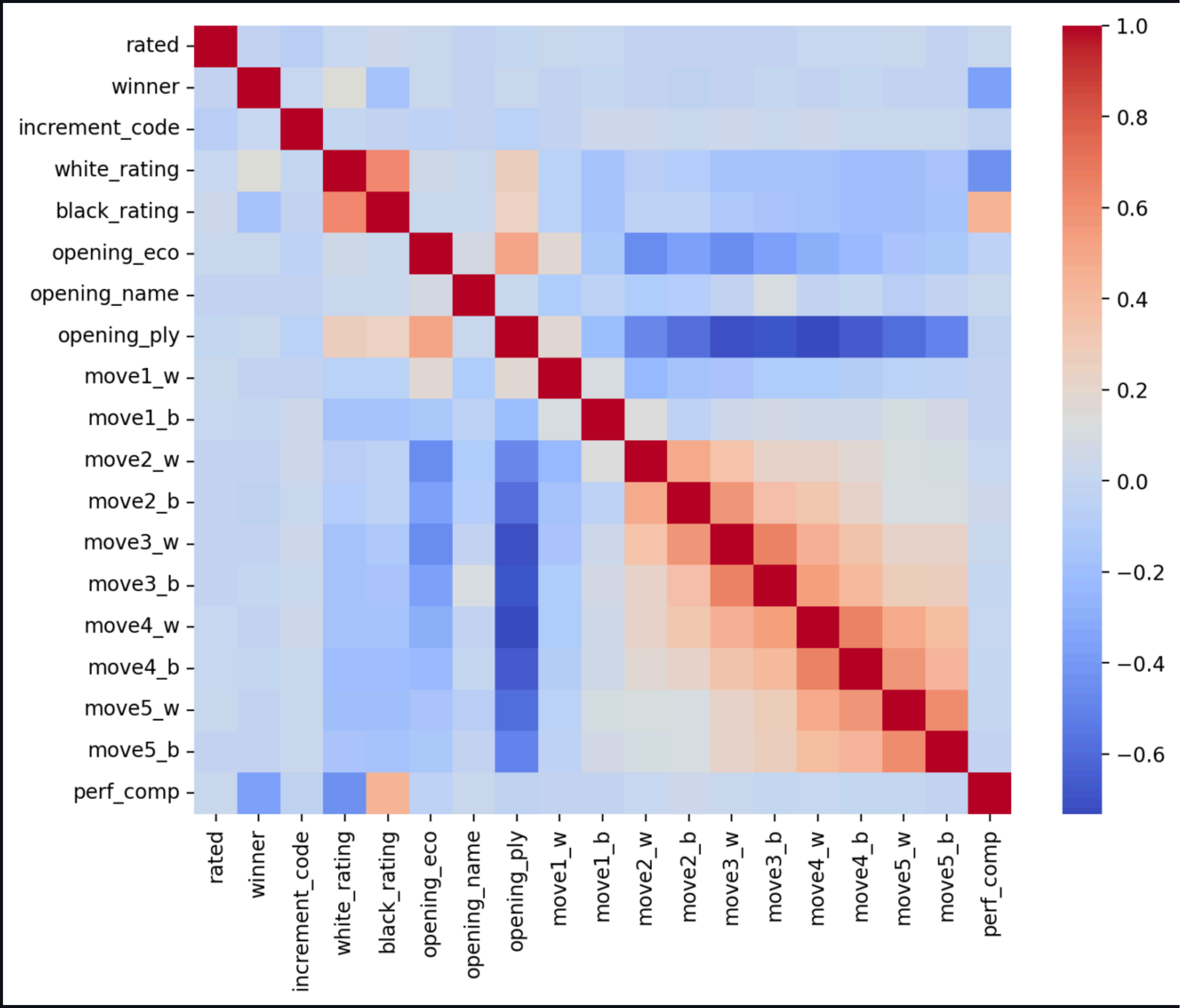
XGBoost

XGBoost (eXtreme Gradient Boosting) is a supervised learning algorithm known for handling large datasets and reducing overfitting. After preprocessing, Grid Search Cross-Validation tuned hyperparameters to optimize performance.

4. Results and Discussion

Visualizations

Correlation Matrix



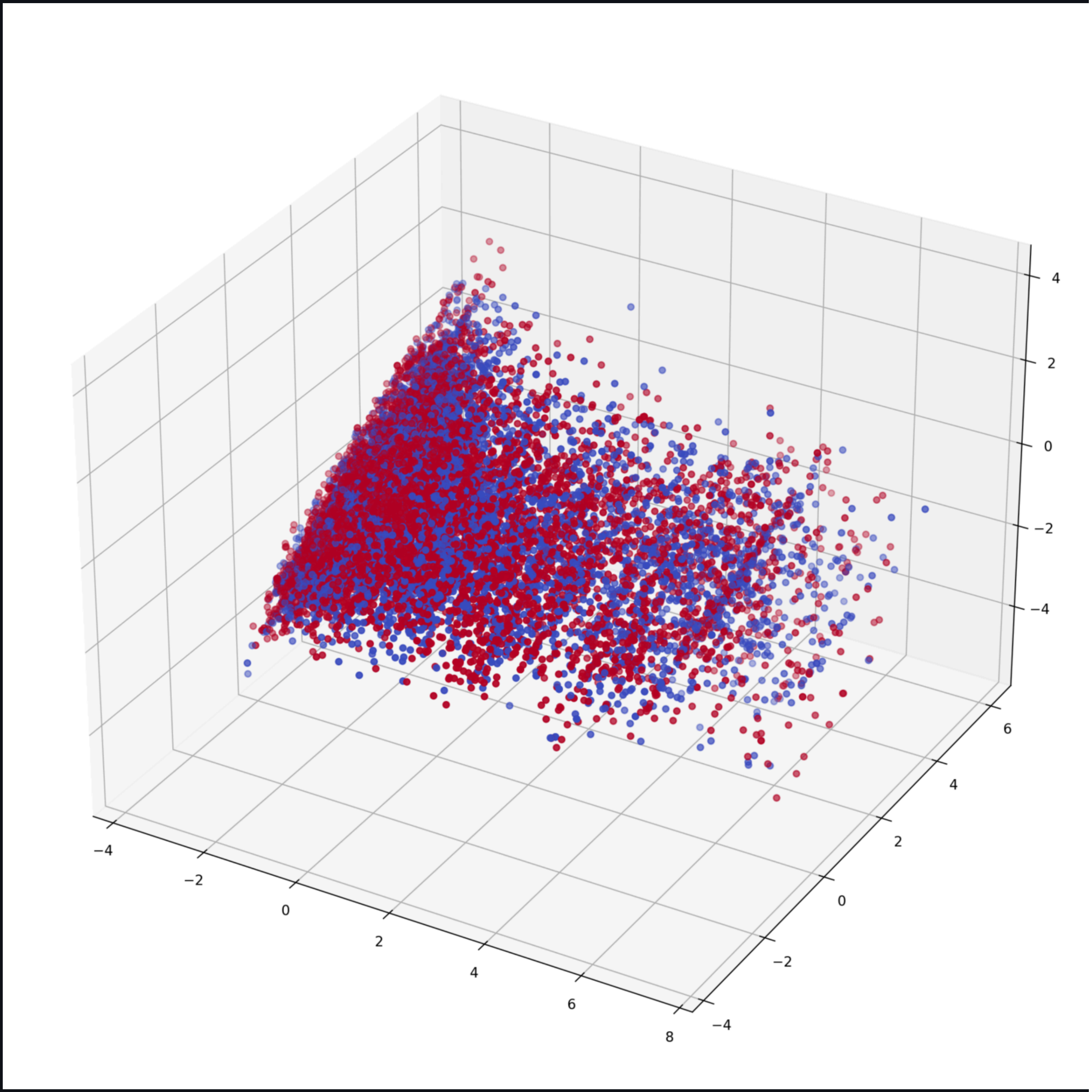
Insights from Correlation Matrix

Highly Correlated Features

- `move2_b`, `move3_b`, `move4_b`, and `move5_b` are strongly correlated, indicating sequential patterns or dependencies in gameplay.

- `white_rating` and `black_rating` are moderately correlated, reflecting that higher-rated players often face one another.

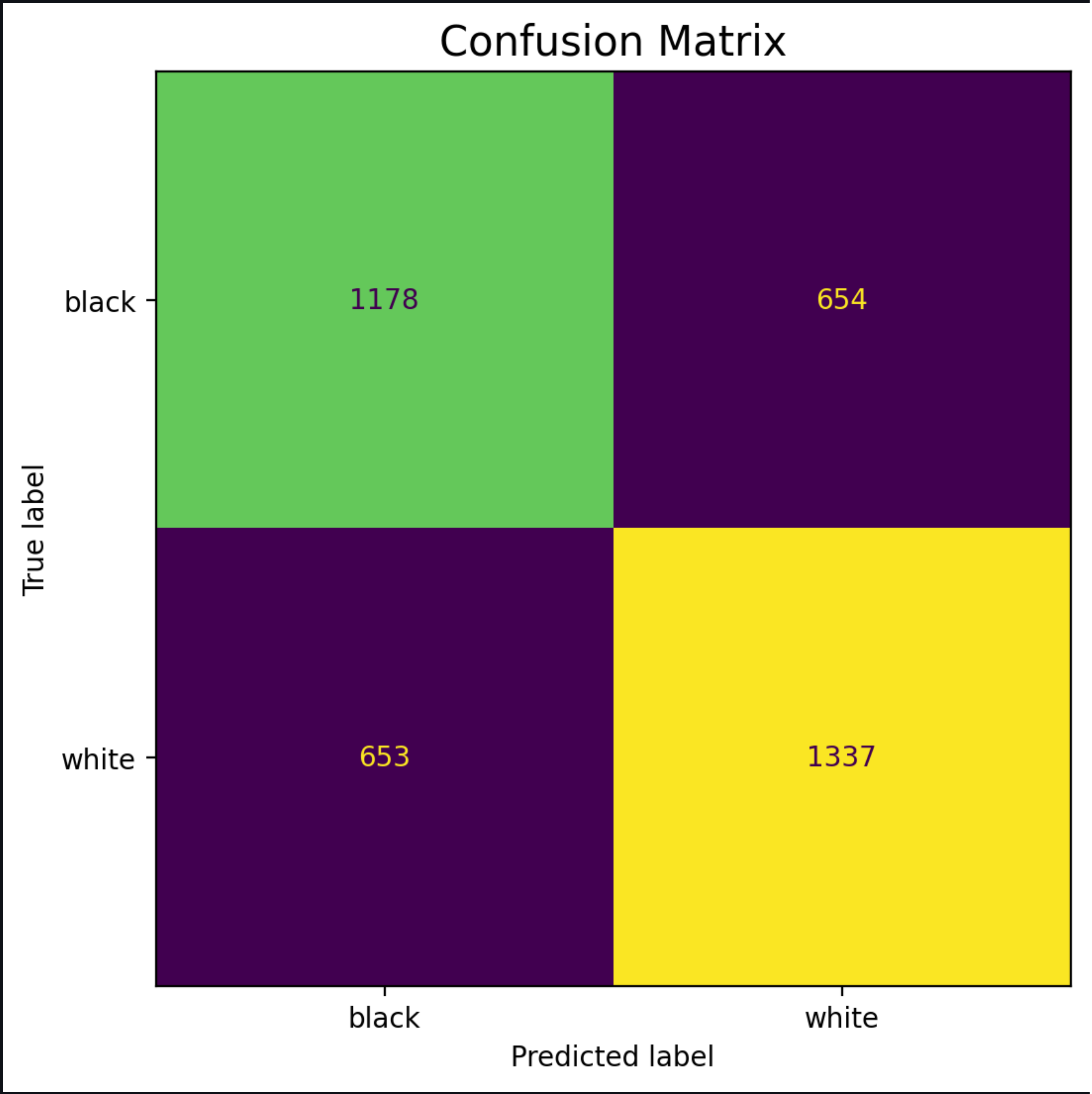
PCA Scatter Plot



Insights from PCA Scatter Plot

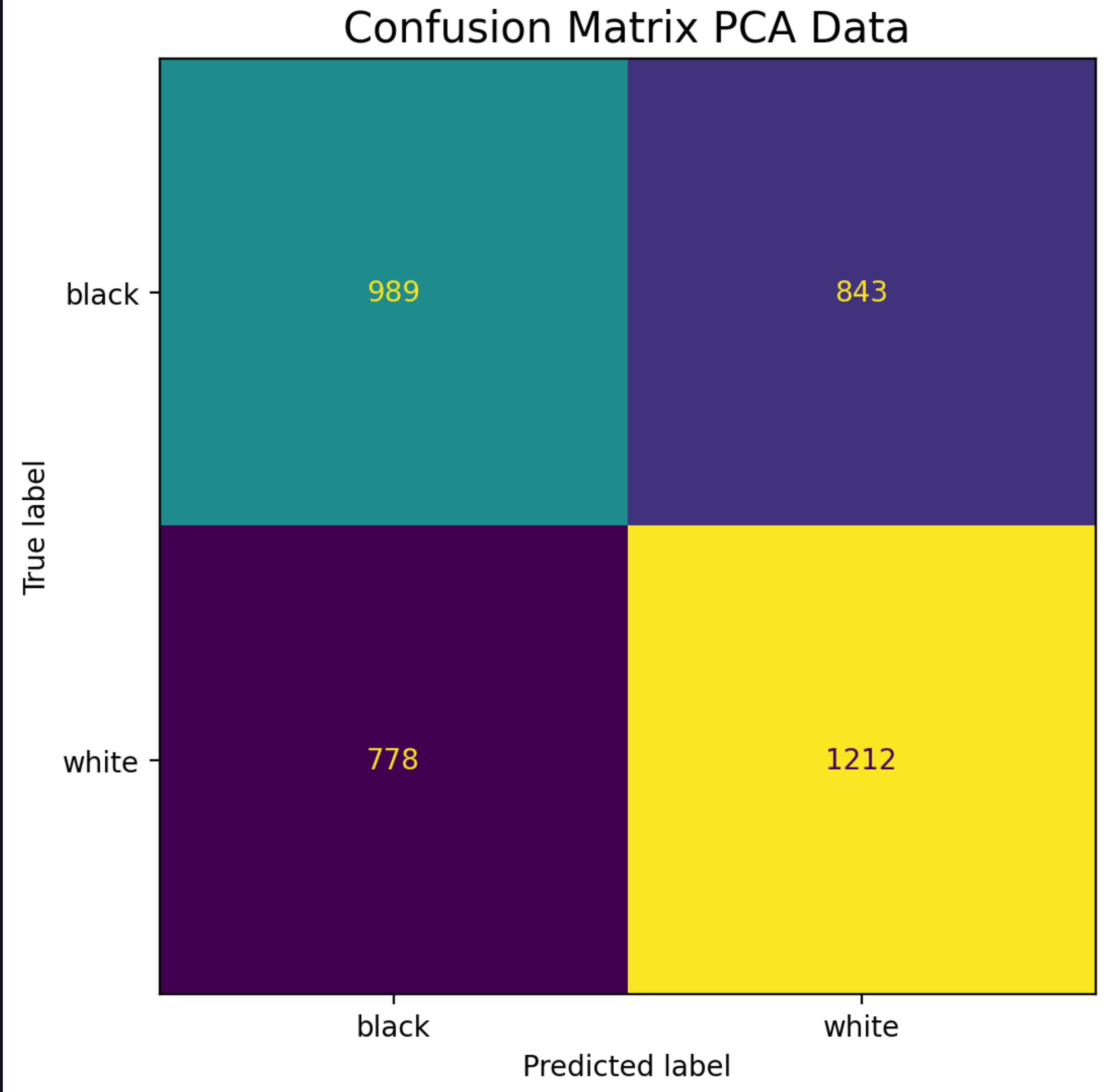
Overlap between classes suggests limited separation using only three principal components. While PCA reduced the dataset to three dimensions for visualization, retaining more components may improve class separability.

Random Forest



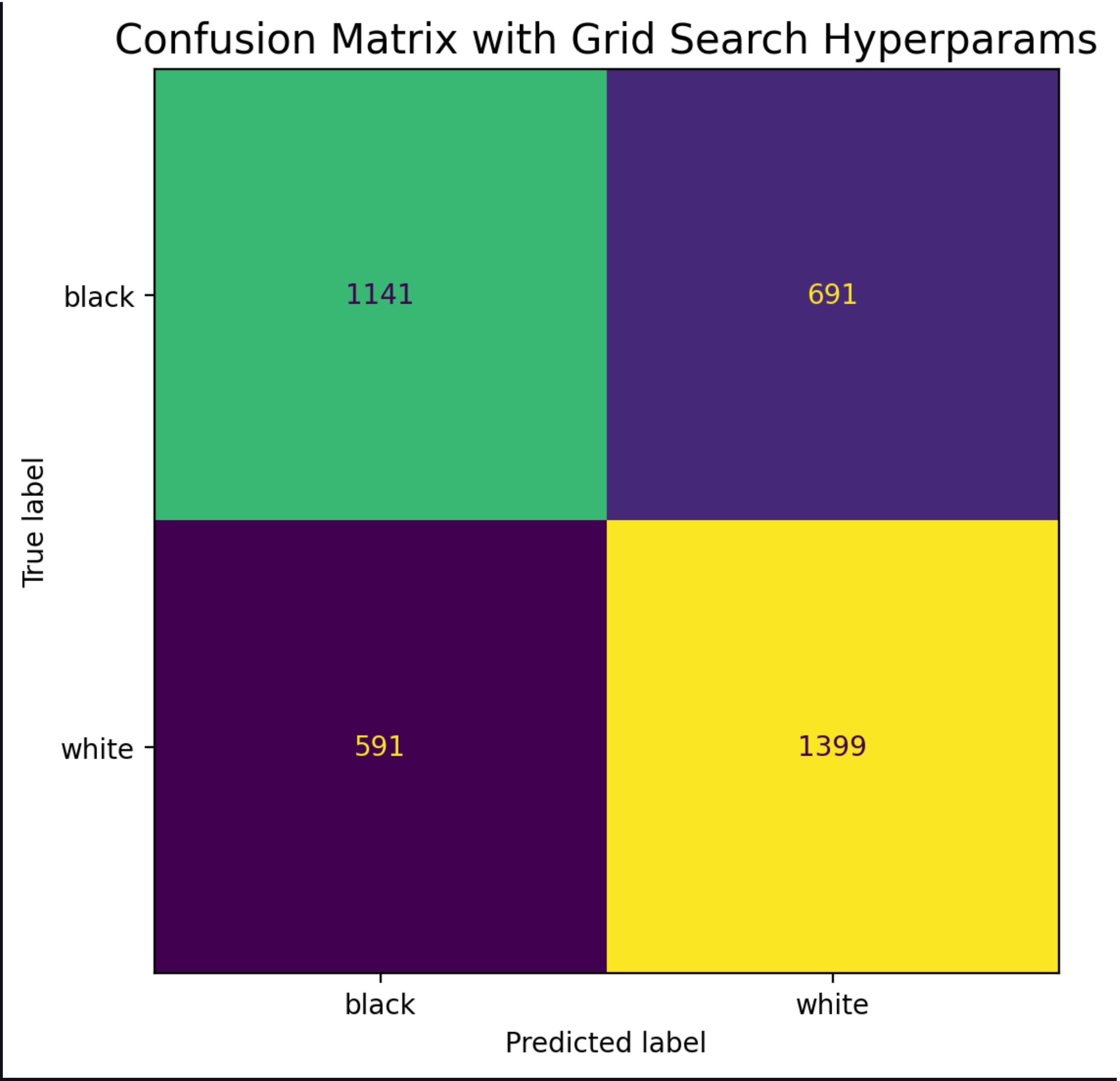
Insights:

- The model performs slightly better at predicting white wins compared to black wins.
- There are more false positives and false negatives for black, suggesting room for improvement.



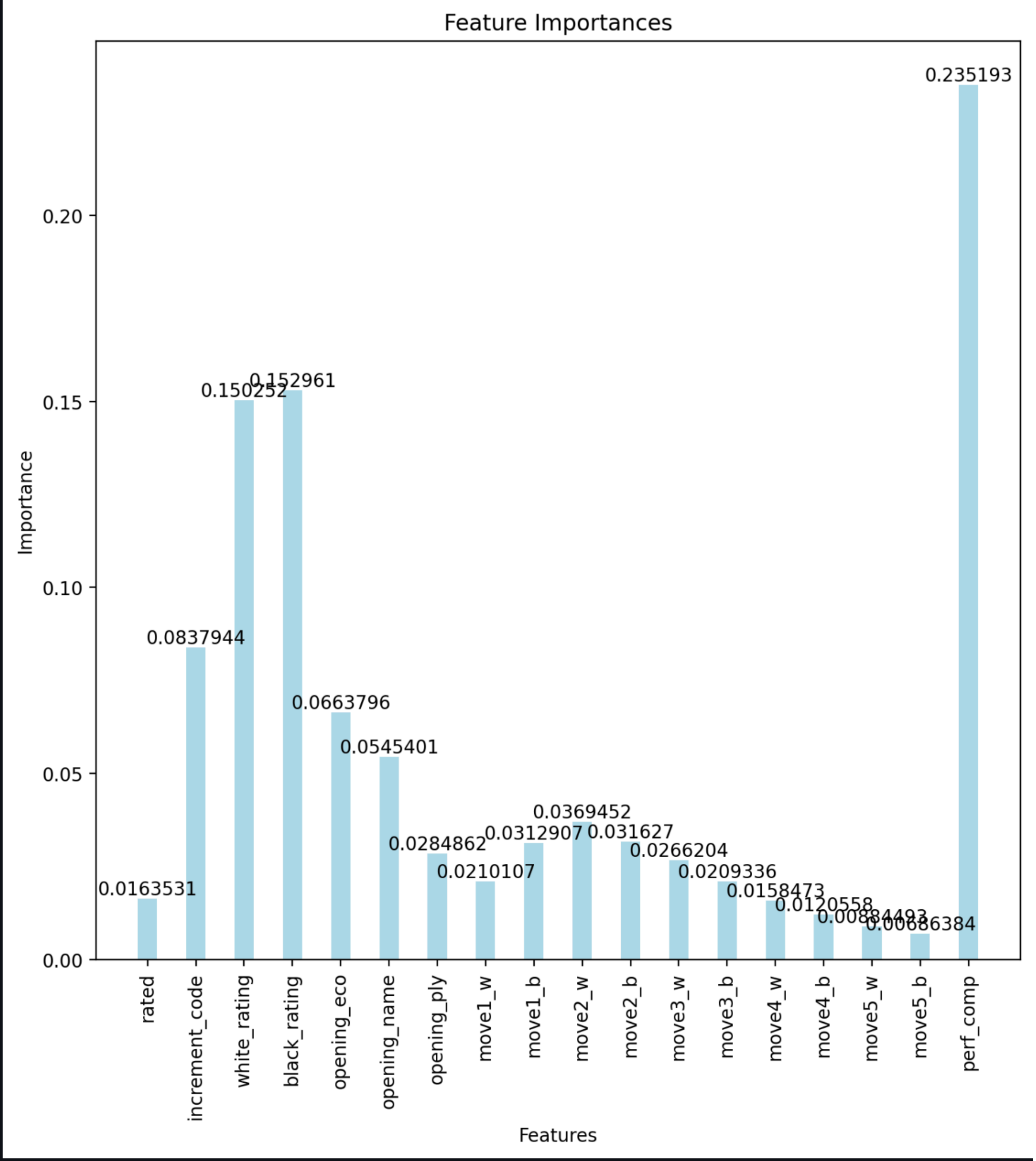
Insights:

- This model underperforms compared to the basic Random Forest model.
- PCA may have removed important features, leading to higher misclassifications.



Insights:

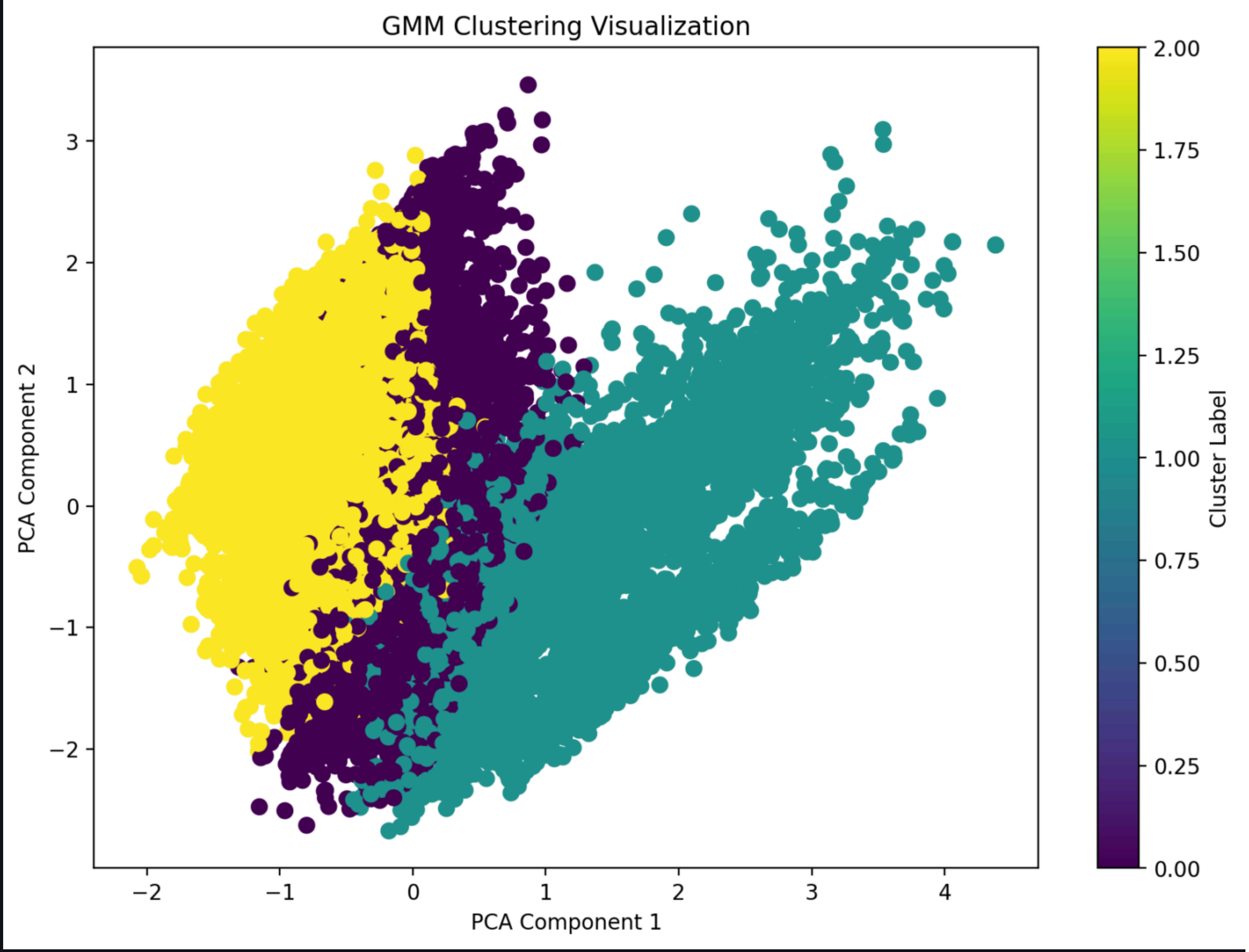
- This model demonstrates the best balance between precision and recall for both classes.
- The benefits of hyperparameter tuning are evident in the reduced misclassifications.



Insights:

- Features like `perf_comp`, `white_rating`, and `black_rating` are the most influential for predictions.

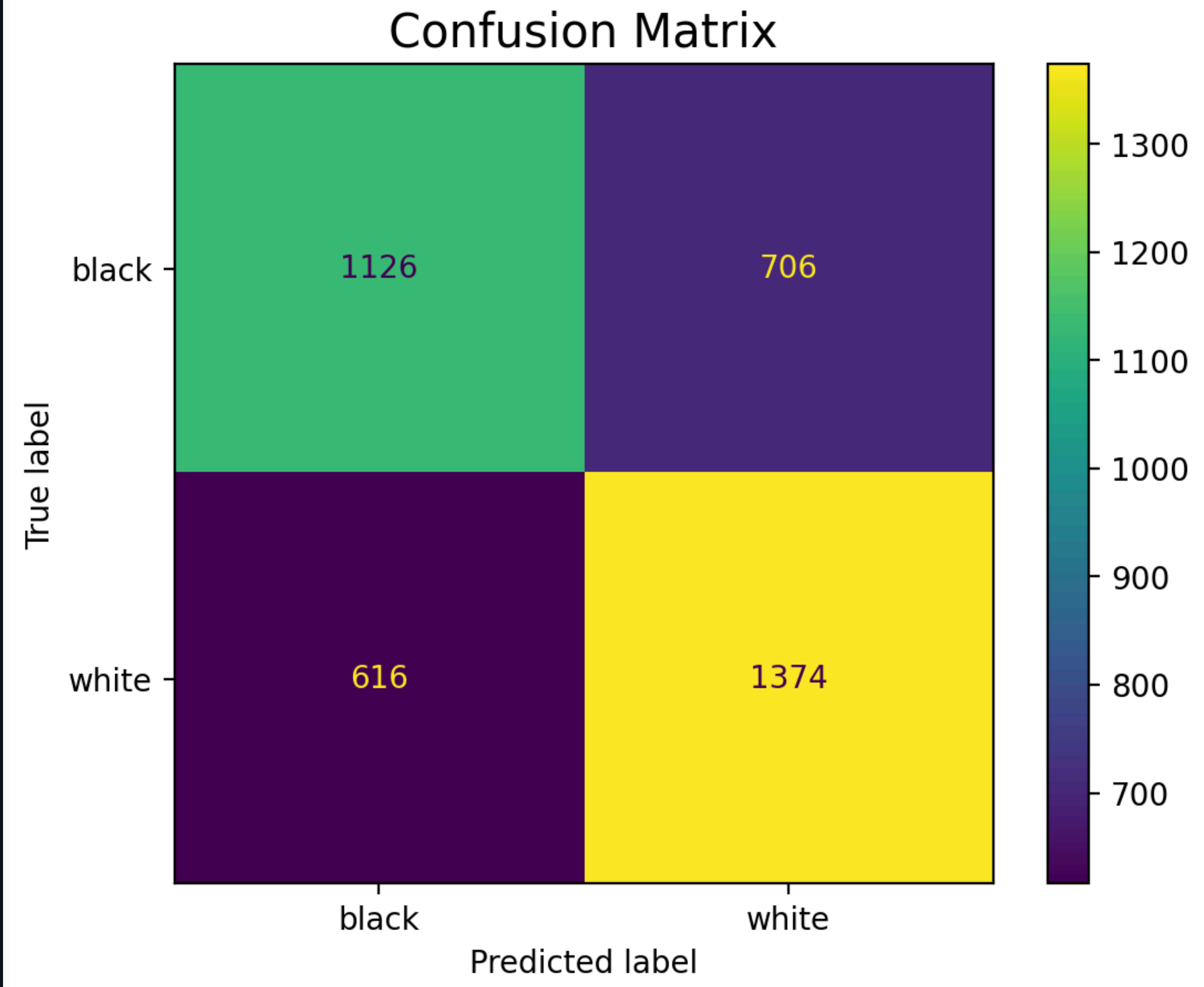
GMM Clustering Visualization



Evaluation:

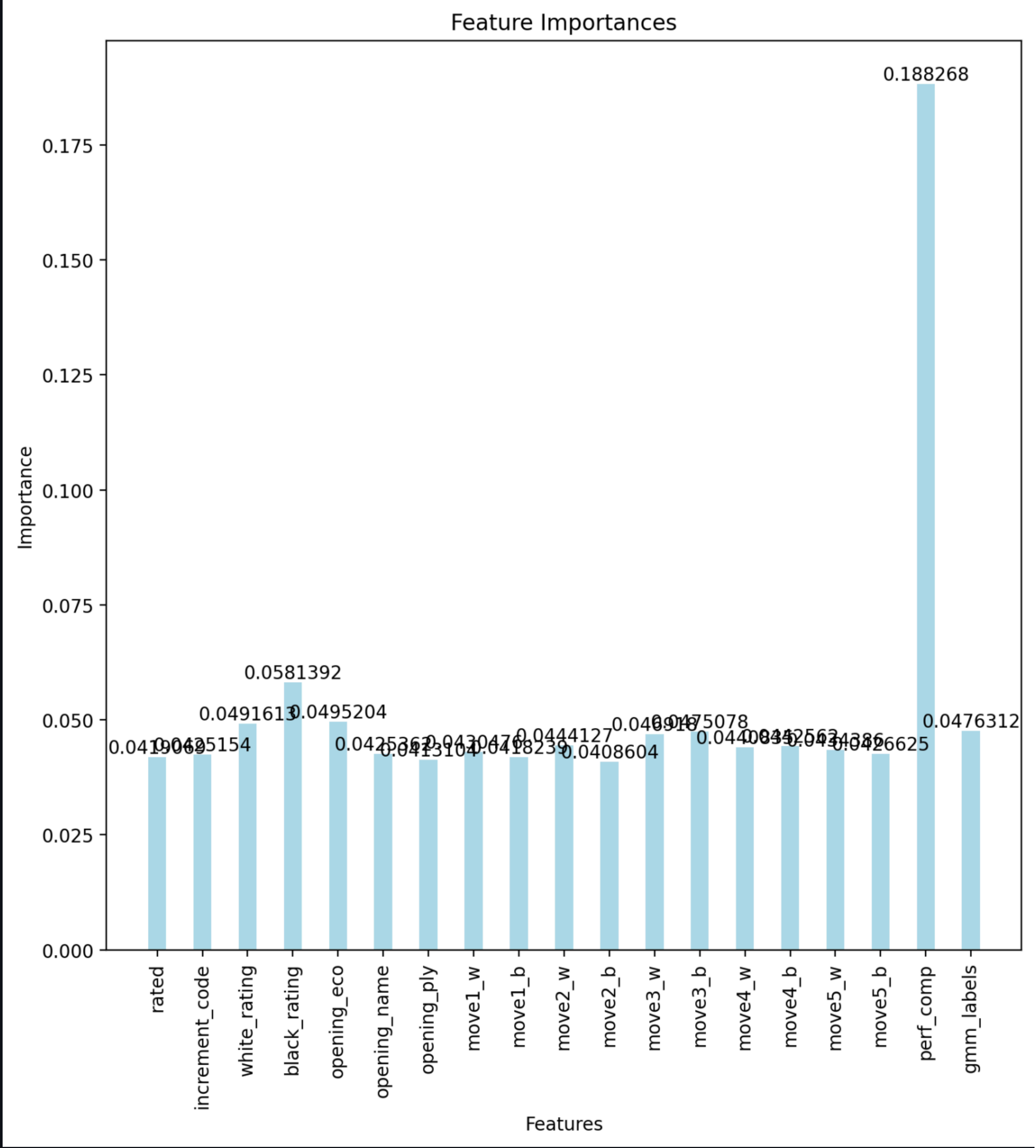
- The GMM clustering visualization shows the dataset distributed across three clusters, identified by colors.
- PCA was used to reduce dimensionality to two components for visualization purposes, making it easier to interpret the cluster distribution.
- The clustering provides meaningful separations, where some overlap between clusters may represent games with closer outcomes or more ambiguous patterns.

XGBoost



Insights:

- The model achieves a good balance between precision and recall for both classes.
- False negatives are slightly fewer than false positives, showing the model has a slight bias toward predicting "white" outcomes.



Evaluation:

- The most important feature is `perf_comp`, which significantly impacts the prediction of chess outcomes.
- Other key features include `white_rating`, `black_rating`, and `increment_code`, highlighting the importance of player ratings and game settings.

Quantitative Metrics and Results

Random Forest

The Random Forest Classifier was used for binary classification to predict chess game outcomes. It offers high accuracy and robustness by using an ensemble of decision trees to capture non-linear patterns.

We implemented three models:

1. A basic model using a manually reduced dataset.
2. A model trained on a PCA-reduced dataset.
3. A model trained with optimized hyperparameters.

Basic Model

Train Accuracy: `0.9990841292686118`

Test Accuracy: `0.6580324437467294`

K-Fold Cross Validation Results:

Mean Accuracy: **65.85%**

Accuracies:

```
[
  0 : 0.6658
  1 : 0.6632
  2 : 0.656
  3 : 0.6409
  4 : 0.6573
  5 : 0.6534
  6 : 0.6649
  7 : 0.6492
  8 : 0.676
  9 : 0.6584
]
```

	Black	White	Overall
Precision	0.6400	0.6700	
Recall	0.6300	0.6800	
F1-Score	0.6400	0.6700	Accuracy: ~66%
Support	1,832.0000	1,990.0000	3822

Analysis:

- Precision for "white" (0.67) is slightly higher than "black" (0.64), meaning the model predicts "white" more accurately.
- Recall for "white" (0.68) is also higher, indicating better coverage for predicting "white" wins.
- The overall accuracy of 65.85% shows the model is moderately effective, with room for improvement, particularly in balancing predictions for both classes.

PCA-Reduced Dataset

Train Accuracy: `0.9990841292686118`

Test Accuracy: `0.575876504447933`

K-Fold Cross Validation Results:

Mean Accuracy: **56.53%**

Accuracies:

```
[
  0 : 0.5605
  1 : 0.5644
  2 : 0.5612
  3 : 0.5683
  4 : 0.55
  5 : 0.5566
  6 : 0.5713
  7 : 0.5759
  8 : 0.5818
  9 : 0.5628
]
```

	Black	White	Overall
Precision	0.5600	0.5900	
Recall	0.5300	0.6200	
F1-Score	0.5500	0.6000	Accuracy: ~58%
Support	1,832.0000	1,990.0000	3822

Analysis:

- The drastic drop in accuracy to 56.53% suggests that PCA removed key features critical for prediction.
- Precision and recall for "black" (0.56 and 0.53) are particularly low, showing the model struggles more with this class.
- PCA may not have been suitable for this dataset, indicating that dimensionality reduction techniques need careful evaluation.

Best Hyperparameters

Train Accuracy: `0.964215622137904`

Test Accuracy: `0.6645735217163788`

K-Fold Cross Validation Results:

Mean Accuracy: **66.23%**

Accuracies:

```
[
  0 : 0.6763
  1 : 0.6566
  2 : 0.654
  3 : 0.6494
  4 : 0.6651
  5 : 0.6573
  6 : 0.6721
  7 : 0.6551
  8 : 0.6747
  9 : 0.6623
]
```

	Black	White	Overall
Precision	0.6500	0.6700	
Recall	0.6200	0.7000	
F1-Score	0.6400	0.6800	Accuracy: ~66%
Support	1,832.0000	1,990.0000	3822

Analysis:

- The highest overall accuracy (66.23%) demonstrates the effectiveness of hyperparameter tuning.
- The model achieves more balanced predictions, with only slight bias toward predicting "white" wins.
- This configuration is the most effective Random Forest model tested.

Gaussian Mixture Model (GMM)

GMM Cluster Results:

	PC1	PC2	PC3	Winner	GMM Labels
0	-0.3062	-1.38	-0.0488	1	1
1	-1.6399	0.2194	0.2051	0	1
2	-1.5331	-0.1037	0.4386	1	1
3	-1.1133	-0.5826	-0.2549	1	1
4	-0.2553	-1.1488	-0.2222	1	0

GMM clustering effectively identifies underlying patterns in the data, which can enhance prediction accuracy when used as features in supervised models. This shows the potential of combining unsupervised learning with supervised approaches.

XGBoost

Grid Search Results:

	Value
colsample_bytree	0.8
learning_rate	0.1
max_depth	4
n_estimators	200
subsample	0.6

Train Accuracy: `0.8287975925683632`

Test Accuracy: `0.6541077969649398`

K-Fold Cross Validation Results:

Mean Accuracy: **0.65**

Accuracies:

```
[
  0 : 0.6566
  1 : 0.6534
  2 : 0.6364
  3 : 0.6331
  4 : 0.6455
  5 : 0.637
  6 : 0.6636
  7 : 0.644
  8 : 0.6702
  9 : 0.6505
]
```

	Black	White	Overall
Precision	0.6400	0.6600	
Recall	0.6000	0.7000	
F1-Score	0.6200	0.6800	Accuracy: ~65%
Support	1,832.0000	1,990.0000	3822

Analysis:

- XGBoost is competitive with the best-tuned Random Forest model, achieving balanced metrics across classes.
- It performs slightly worse in predicting "black" outcomes but maintains high overall performance due to robust optimization.

Analysis and Comparison of Models

Model Performance:

- Random Forest with Hyperparameter Tuning achieved the highest accuracy (~66%) and balanced precision/recall.
- XGBoost performed slightly lower at (~65%) accuracy but was efficient in handling large datasets.
- PCA-Reduced Random Forest demonstrated the weakest performance, highlighting the importance of feature engineering.

Key Insights:

- Both Random Forest and XGBoost identified `perf_comp`, `white_rating`, and `black_rating` as the most important features.
- GMM clustering revealed underlying patterns in the data, which, when incorporated into supervised models, enhanced prediction accuracy.
- Hyperparameter optimization improved both Random Forest and XGBoost performance, reducing false positives and false negatives.

Next Steps

1. **Feature Engineering:**
    - Focus on enhancing high-importance features such as `perf_comp` and player ratings.
2. **Model Optimization:**
    - Explore combining Random Forest and XGBoost.
    - Experiment with advanced hyperparameter optimization using Bayesian optimization.
3. **Deep Learning Exploration:**
    - Investigate deep learning models like LSTMs or Transformers for sequence data.

5. References

[1] I. Cheng, “Machine Learning to Study Patterns in Chess Games,” Mar. 2024, Accessed: Oct. 04, 2024. [Online]. Available: https://www.researchgate.net/profile/Isaac-Cheng-3/publication/379334134_Machine_Learning_to_Study_Patterns_in_Chess_Games/links/6604c292390c21 4cfd151950/Machine-Learning-to-Study-Patterns-in-Chess-Games.pdf

[2] K. Raghav and L. Ahuja, "Chess Opening Analysis Using DBSCAN Clustering and Predictive Modeling," 2024 11th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2024, pp. 1-5, doi: 10.1109/ICRITO61523.2024.10522439.

[3] D. Silver, T. Hubert, J. Schrittwieser, and D. Hassabis, “AlphaZero: Shedding new light on chess, shogi, and Go,” Google DeepMind, Dec. 06, 2018. https://deepmind.google/discover/blog/alphazero-shedding-new-light-on-chess-shogi-and-go/ (accessed Oct. 04, 2024).

Gantt Chart

Group 5 Gantt Chart:
https://docs.google.com/spreadsheets/d/1CYhvUaiTCBzN0lFuOxkwd5xmzU_AKQHE/edit?usp=sharing&ouid=111692925742591404356&rtpof=true&sd=true

Contribution Table

	Name	Contribution
0	Bryson Bien	ML Model, Methods, Slides , ReadMe
1	Marcel Dunat	Preprocessing, Methods, Slides
2	Harrison Speed	Quantitative Metrics, Visualizations, Slides
3	Noah Brown	ML Model, Methods, Slides
4	Patrisiya Rumyantseva	Discussion/Comparison, Streamlit, Video, Slides

Google Slides

https://docs.google.com/presentation/d/1vxEmXyQK5AcFjSmRBrUk0hyt9E6KFblMp8662HgS4QM/edit?usp=sharing