Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control
Journal homepage: http://kinetik.umm.ac.id
ISSN: 2503-2267
Vol. 11, No. 3, August, Pp. 559-570

Dota 2 hero buff and nerf predictions based on professional match data using random forest

Muhammad Raditya Azanata[1], Muhamad Azrino Gustalika[*1], Dimas Fanny Hebrasianto Permadi[1]
Telkom University Purwokerto, Indonesia[1]

Article Info
Keywords:
Dota 2, Game Balancing, Random Forest, Patch Analysis, Recommendation System

Article history:
Received: January 21, 2026
Accepted: May 16, 2026
Published: August 01, 2026

Cite:
M. Raditya Azanata, M. Azrino Gustalika, and D. Fanny Hebrasianto Permadi, "Dota 2 Hero Buff and Nerf Predictions Based on Professional Match Data Using Random Forest", *KINETIK*, vol. 11, no. 3, Aug. 2026. https://doi.org/10.22219/kinetik.v11i3.2698

*Corresponding author.
Muhamad Azrino Gustalika
E-mail address:
azrino@telkomuniversity.ac.id

Abstract
Balancing updates (buffs and nerfs) are essential in Multiplayer Online Battle Arena games because minor parameter changes can shift the competitive metagame and reduce hero diversity. Unlike previous studies on Dota 2 that focus on match outcome prediction, this study introduces and evaluates hero-centric balance recommendations against official patch actions across patch transitions. To address this gap, this work contributes a patch-to-patch (Version 7.39-7.40b) external validation protocol that compares recommendations from patch t with developer actions in patch t+1 using patch notes. Professional match records were gathered from public sources and sorted by hero and patch into combat, economic, and impact categories. This study proposed a data-driven pipeline to classify each Dota 2 hero as overpowered, underpowered, or balanced from professional match telemetry and to translate these classes into balance recommendations (Nerf, Buff, or Balance). Labels derive from win-rate and pick-rate distributions using statistical control limits ($\mu \pm k\sigma$, $k = 0.3$) to ensure transparent, repeatable labeling. A Random Forest classifier was trained using grid-searched hyperparameters and evaluated using stratified 6-fold cross-validation with macro-averaged F1 to address class imbalance. Internal evaluation achieved 0.94 accuracy and 0.84 macro-F1. For external validation, patch t recommendations were compared to official balance actions in patch t+1 during six successive transitions; accuracy ranged from 0.436 to 0.672 (mean 0.559), with the best result on 7.39b to 7.39c (84/125). These results indicated that professional telemetry could support interpretable balance monitoring and provide early signals for buff/nerf candidate review.

1. Introduction

Modern esports ecosystems evolve through frequent balance updates that aim to maintain fairness, diversity, and long-term player engagement [1][2]. In Multiplayer Online Battle Arena (MOBA) games such as Dota 2, even small changes to hero attributes can significantly shift the competitive metagame, affecting drafting priorities, role viability, and strategic outcomes [3][4][5]. However, these approaches do not directly address how hero-level performance translates into balancing decisions, nor do they evaluate alignment with developer actions across patch transitions. As a result, their applicability to practical game balancing remains limited [3][6][7][8]. While these studies demonstrate the usefulness of gameplay telemetry, they typically operate at the match level and do not directly address hero-centric balance analysis or its relationship to developer decisions across game patches [10][11][12][13][14][15].

Despite the availability of performance indicators such as win rate and pick rate, there is currently no standardized, scalable framework for reliably interpreting these metrics to support balanced decision-making. This leaves a gap between raw data signals and actionable balancing recommendations[5][9]. Common indicators such as win rate and pick rate provide useful signals of hero performance and popularity, but interpreting them in a consistent, scalable framework remains challenging [1][2]. Furthermore, the dynamic nature of game patches introduces distribution shifts, necessitating evaluation of models not only internally but also across patch transitions [5][9]. Moreover, frequent patch updates introduce distribution shifts and structural gameplay changes, making it challenging for models trained on one patch to generalize effectively to subsequent patches [3][20][24]. Therefore, evaluation strategies should not rely solely on internal validation but also consider cross-patch generalization to reflect better real-world deployment scenarios [10][12][14]. To the best of our knowledge, this study is among the first to frame hero balancing as a supervised classification problem with explicit patch-to-patch external validation against official developer decisions. Unlike prior work that focuses on match-level prediction, this research introduces a hero-centric perspective that aligns directly with practical game-balancing objectives.

This study addresses these gaps by proposing a data-driven framework for hero balance classification and recommendation using professional match telemetry. The approach formulates hero balancing as a multi-class

classification problem (Overpowered, Underpowered, Balanced) and translates predictions into actionable recommendations (Nerf, Buff, or Balance).

This study makes several key contributions; firstly, it proposes a hero-centric balance classification framework based on aggregated professional match data. In addition, it introduces a patch-to-patch external validation protocol that allows comparison between model recommendations and official developer actions across consecutive patches; develops a transparent statistical labeling scheme ($\mu \pm k\sigma$) using win rate and pick rate distributions, and provides robustness analysis encompassing sensitivity analysis of the threshold parameter, class imbalance handling, and data leakage evaluation.Experimental results show strong internal performance and moderate external agreement with developer decisions, indicating that professional match telemetry can provide useful early signals for balance monitoring. However, this study also highlights that developer balancing decisions are influenced by broader gameplay and design considerations beyond statistical performance. Therefore, this study serves as a decision-support tool to help identify potential heroes in need of further balance review, not a substitute for developer judgment.

2. Research Method

The overall workflow of the proposed approach present in Figure 1. The pipeline starts from professional match data and official patch notes, followed by preprocessing, per-hero per-patch aggregation, feature construction, threshold-based labeling (OP/UP/Balanced), Random Forest training, internal evaluation using stratified 6-fold cross-validation, and patch-to-patch external validation against patch-note actions.

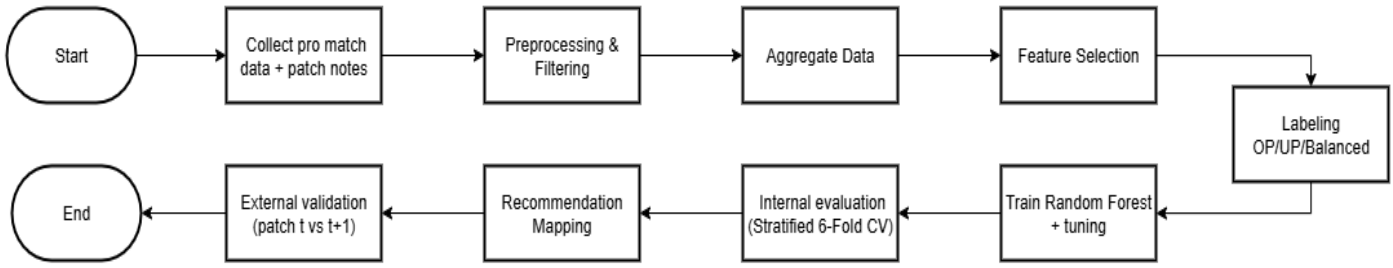

Figure 1. Workflow

2.1 Data Collection and Unit of Analysis

This study utilizes two main data sources: (1) professional match data obtained from public APIs, and (2) official patch notes used as ground truth for external validation.

The raw match dataset contains per-match statistics for each hero, including combat, economy, and impact metrics such as kills, deaths, assists, damage dealt, tower damage, healing, gold per minute (GPM), experience per minute (XPM), and match outcomes (win/loss). Each row represents a hero's performance in a single match. Table 1(a) presents an example of the raw Dataset before preprocessing.

2.1.1 Raw Match Telemetry from API

Professional match data collect at the hero–match level, where each row corresponds to a hero's appearance in a single professional match. For a patch containing 611 matches, the raw match in table 1 contains approximately 6110 rows. The raw fields extracted from the API are shown in Table 1, including hero_id, localized_name, kill, death, assist, damage_dealt, tower_damage, healing_done, gold_per_min, xp_per_min, and the match outcome indicator win. Raw telemetry fields are widely used in esports analytics and outcome prediction because they capture combat efficiency, progression, and impact signals.

Table 1. Raw Match-Level Dataset

match_id	hero_id	localized_name	total_matches	kill	death	assist	damage_dealt	tower_damage	healing_done	gold_per_min	xp_per_min	win
8605793716	14	Pudge	611	4	12	7	8732	0	0	264	396	0
8605793716	38	Beastmaster	611	11	9	3	24133	1927	0	599	722	0
8605793716	39	Queen of Pain	611	12	4	10	22196	4319	0	627	911	1
8605793716	40	Venomancer	611	7	7	25	22457	1398	1673	463	728	1

2.1.2 Patch Notes for Ground Truth (External Validation)

Patch notes provide a reference for the actions developers took (buffs or nerfs) and enable patch-to-patch evaluation [1][3][22]. For external validation, we manually transform patch notes into a compact hero-level table (Table 2) consisting of two columns: localized_name and patch_action, where patch_action ∈ {Buff, Nerf, Balance}. Heroes not explicitly mentioned in the notes are treated as Balance to ensure full coverage during evaluation. In addition, we collect patch notes and convert them into a structured dataset containing hero names and corresponding patch actions (Buff, Nerf, or Balance) for external validation.

Table 2. Data Ground Truth

localized_name	*patch_action*
Witch Doctor	*Nerf*
Wraith King	*Nerf*
Batrider	*Buff*
Beastmaster	*Buff*

2.2 Preprocessing and Filtering

Preprocessing ensures reliable integration between telemetry records and patch-note labels in subsequent stages. First, we remove incomplete telemetry rows with missing key combat, economy, or impact statistics using predefined rules, and standardize all numeric columns into consistent data types to avoid aggregation errors [16][17][23]. Second, we standardize hero identifiers using hero_id and localized_name to ensure consistent merging between the aggregated Dataset and the patch-note table at the hero level. Although Random Forest does not require feature scaling, maintaining consistent numeric formats enhances reproducibility and reduces data noise [16][17]. Finally, we segment the filtered telemetry records by patch to enable per-patch aggregation and evaluation under patch-driven distribution shifts. [3].

2.3 Aggregation: Hero–Patch Dataset

After preprocessing, we aggregate hero match telemetry into a hero–patch dataset to align with the granularity of balance decisions, which are inherently centered at the hero level [1][24]. We perform aggregation independently for each patch, producing one row per hero. For each hero, we summarize telemetry features using the mean across all matches in which the hero appears within the patch. We further derive exposure-based variables, including pick (number of unique matches), wins (number of matches won), $win_{rate} = \frac{wins}{pick}$, and $pick_{rate} = \frac{pick}{total_{matches}}$. This aggregation reduces match-level variability and noise while preserving representative performance signals at the patch level. The resulting Dataset is presented in Table 3.

Because the hero roster is fixed in the evaluated period, each patch-level aggregated table contains 126 heroes, allowing for stable comparisons across successive patches and 14 feature columns (Table 3), plus one label column (OP/UP/Balanced) defined in the next section. This compact representation supports stable training and comparison across consecutive patch transitions while capturing meta shifts through pick_rate and win_rate [1][3][25]. After preprocessing, the Dataset is aggregated from per-match to per-hero level for each patch. The aggregation computes the average value of each feature across all matches and derives additional variables such as pick and pick_rate. Overall, the hero-patch representation provides a compact and stable feature space that supports reliable model training and meaningful cross-patch generalization under evolving game conditions.

Table 3. Example of Aggregated Hero Dataset (after preprocessing)

id	*Hero_name*	*pick*	*win_rate*	*kill*	*death*	*assist*	*damage_dealt*	*tower_damage*	*healing_done*	*gold_per_min*	*xp_per_min*	*total_matches*	*pick_rate*
1	*Anti-Mage*	*31*	*0.3548*	*8.71*	*5.26*	*6.77*	*19525.55*	*4322.19*	*11.06*	*623.74*	*821.90*	*611*	*0.0507*
2	*Axe*	*102*	*0.4608*	*7.69*	*6.72*	*9.74*	*19339.77*	*764.58*	*3.68*	*504.38*	*640.82*	*611*	*0.1669*
3	*Bane*	*17*	*0.5882*	*2.29*	*6.35*	*15.94*	*9533.47*	*205.41*	*49.59*	*289.35*	*448.35*	*611*	*0.0278*
4	*Bloodseeker*	*20*	*0.6500*	*9.50*	*5.25*	*10.85*	*29489.50*	*4362.60*	*72.80*	*630.15*	*829.80*	*611*	*0.0327*
5	*Crystal Maiden*	*78*	*0.5513*	*2.79*	*7.72*	*15.38*	*11869.56*	*307.13*	*66.62*	*311.46*	*465.26*	*611*	*0.1277*

2.4 Feature Selection (Model Inputs)

We perform feature selection after aggregation, as model inputs are defined on the hero–patch representation, in accordance with common practices in esports telemetry modeling [2][6][7], We group candidate features into: (i) combat (kills, deaths, assists), (ii) economy/progression (GPM, XPM), (iii) impact (damage dealt, tower damage, healing

done), and (iv) exposure (pick_rate, win_rate). We select these features as they capture complementary dimensions of hero performance, including combat effectiveness, resource accumulation, and overall contribution to match outcomes. Random Forest can handle mixed feature scales and non-linear interactions effectively, making it suitable for structured aggregated telemetry [8], [26]. However, it is important to note that win_rate and pick_rate are also used in the statistical labeling process. To mitigate potential data leakage, this study explicitly evaluates model robustness by conducting additional experiments that exclude these variables from the feature set. This design ensures that the model does not rely solely on variables that directly define the target labels, but can still learn meaningful patterns from combat, economy, and impact features. In this study, the final feature set includes the aggregated combat/economy/impact indicators and the exposure rates (win_rate, pick_rate) because these directly reflect both performance and meta prevalence, which are central to balance analysis [24]. Therefore, the inclusion of exposure features is treated as a design choice rather than an assumption, and its effect on model performance is systematically analyzed in the robustness evaluation.

2.5 Statistical Labeling of Hero Status

To construct the target classes for supervised learning, each hero is assigned one of three balance-status labels: Overpowered (OP), Underpowered (UP), or Balanced, based on aggregated hero performance within a patch. This labeling strategy uses a transparent statistical thresholding rule derived from the distribution of win rate and pick rate, which jointly capture hero effectiveness and popularity in competitive play [1], [2], [22]. By combining these two indicators, we ensure that the labeling process captures both hero performance and consistent preference in competitive play.

The threshold boundaries are determined using the mean and standard deviation of hero-level distributions within each patch. First, the upper win-rate threshold is defined using Equation 1, where μ_{win}represents the average hero win rate and σ_{win}represents its standard deviation. This threshold identifies heroes whose win rate is significantly above the patch average.

$$win_{high} = \mu_{win} + k * \sigma_{win} \quad (1)$$

Second, the lower win-rate threshold is defined using Equation 2, which identifies heroes performing significantly below the patch-wide average.

$$win_{low} = \mu_{win} - k * \sigma_{win} \quad (2)$$

Third, we quantify hero popularity using pick rate and determine its upper threshold according to Equation 3, where $\mu_{pick}$is the mean hero pick rate and $\sigma_{pick}$is its standard deviation. Heroes above this threshold are considered substantially more popular than average.

$$pick_{high} = \mu_{pick} + k * \sigma_{pick} \quad (3)$$

Similarly, Equation (4) defines the lower pick-rate threshold, which identifies heroes selected substantially less frequently than the patch norm.

$$pick_{low} = \mu_{pick} - k * \sigma_{pick} \quad (4)$$

Based on these four thresholds, we assign hero labels according to the decision rules defined in Table 4. A hero is labeled OP when both win rate and pick rate exceed their upper thresholds, indicating dominant performance and strong competitive adoption. Conversely, a hero is labeled UP when both metrics fall below their lower thresholds, suggesting weak performance and limited usage. We categorize heroes that do not meet any extreme conditions as Balanced. This formulation provides a transparent and reproducible labeling mechanism that dynamically adapts to the distribution of each patch. By defining thresholds relative to patch-specific statistics, we capture meta-dependent variations rather than relying on fixed global thresholds, which may not generalize across different game conditions. However, the choice of the threshold coefficient k directly affects class distribution. Smaller values of k result in more heroes being classified as OP or UP, while larger values produce a more conservative labeling dominated by the Balanced class. This introduces inherent class imbalance, which may influence model behavior, particularly in detecting

minority classes (OP and UP). To address this issue, we conduct a sensitivity analysis across multiple values of k (0.1 to 1.0). This analysis evaluates the impact of k on class distribution, internal classification performance, and external validation results across patch transitions.

Table 4. Hero Status Labeling Rule

Label	Rule
OP	win_rate >= win_{high} and pick_rate >= $pick_{high}$
UP	win_rate <= win_{low} and pick_rate <= $pick_{low}$
Balance	otherwise

The threshold coefficient $k$controls the strictness of class separation. Smaller values of $k$generate narrower thresholds, resulting in more heroes being classified as OP or UP, while larger values produce wider thresholds and a more conservative labeling scheme dominated by the Balanced class. Since threshold selection directly influences class distribution and model behavior, we conduct a sensitivity analysis using multiple values of k, with the comparison results presented in Table 5.

Table 5 Sensitivity Analysis of Threshold Coefficient k

k	Internal Accuracy	Macro-F1	External Accuracy
0.1	0.52	0.39	0.608
0.2	0.58	0.40	0.616
0.3	0.65	0.40	0.664
0.5	0.70	0.32	0.672
0.7	0.80	0.33	0.664
1.0	0.95	0.33	0.640

As shown in Table 5, increasing $k$tends to improve internal accuracy because class boundaries become stricter and more observations fall into the dominant Balanced class. However, this comes at the cost of lower macro-F1, indicating weaker recognition of minority classes (OP and UP). Although $k = 0.5$produces the highest external accuracy, the improvement over $k = 0.3$is marginal, while macro-F1 drops substantially. Therefore, $k = 0.3$ is selected in this study because it provides the best trade-off between internal classification quality and external patch-to-patch generalization, while maintaining sensitivity to minority-balanced adjustment classes. In contrast, larger values of k tend to inflate internal accuracy by increasing the dominance of the Balanced class, but reduce the model's ability to detect meaningful balance-adjustment candidates.

2.6 Random Forest Classifier and Hyperparameter Tuning

A Random Forest classifier is used to classify heroes' statuses. This ensemble learning approach creates numerous decision trees using bootstrapped subsets of the training data and aggregates their predictions using majority voting. [8], [16], [26]. At each split, Random Forest randomly selects a subset of features and chooses the split that minimizes node impurity, commonly measured using the Gini impurity criterion. This randomized ensemble design reduces overfitting, improves generalization, and enables the model to capture complex non-linear interactions among features, making it well-suited for structured telemetry data commonly found in game analytics [2], [4], [8]. We select Random Forest because it effectively handles heterogeneous feature types, demonstrates robustness to noise, and requires minimal feature scaling. These properties are particularly advantageous for aggregated gameplay telemetry, where feature distributions may vary across patches and exhibit complex interactions.

We select Random Forest because it effectively handles heterogeneous feature types, demonstrates robustness to noise, and requires minimal feature scaling. These properties are particularly advantageous for aggregated gameplay telemetry, where feature distributions may vary across patches and exhibit complex interactions. The variable $X \in \mathbb{R}^{126\times p}$, where 126 denotes the number of heroes observed in a patch and $p$represents the selected aggregated telemetry features described in Section 2.4, including combat, economy, impact, and exposure indicators. The corresponding target vector is defined as $y \in \{\text{OP}, \text{UP}, \text{Balanced}\}$, where each label is generated using the statistical labeling framework described in Section 2.5. The classifier learns the relationship between aggregated hero performance patterns and hero balance status.

Because the resulting class distribution is naturally imbalanced, with the Balanced class dominating the Dataset, the model incorporates a balanced class weighting scheme $class weight = balanced$ during training. This weighting mechanism assigns higher importance to minority classes (OP and UP) and lower relative importance to the majority

class, allowing the model to learn minority-class decision boundaries more effectively and reducing bias toward predicting only the dominant class [28]. This approach mitigates bias toward majority-class predictions and improves the model's ability to learn meaningful decision boundaries for minority classes, which are critical for balanced recommendation tasks.

To obtain robust model configurations, we perform hyperparameter tuning using grid search within stratified 6-fold cross-validation, ensuring that each fold preserves the original class proportions of OP, UP, and Balanced. The evaluated hyperparameters include the number of trees $n\_estimators$, maximum tree depth $maxdepth$, feature sampling strategy at each split $max\_features$, and minimum samples required at leaf nodes $min\_samples\_leaf$ [24], [25]. Model selection is based on macro-averaged F1-score, since this metric provides balanced evaluation across all classes and prevents majority-class dominance from inflating performance estimates [18]. Overall, the proposed model design is reproducible and flexible, as the same pipeline can be extended to alternative classifiers while preserving the aggregated data representation and evaluation protocol. This makes the framework adaptable for broader applications in game balancing and telemetry-driven decision support.

The best-performing configuration uses 200 decision trees, log2 feature sampling, minimum leaf size of 2, and unlimited tree depth $max\_depth = None$, combined with balanced class weighting. This configuration provides the most stable trade-off between bias and variance while maintaining sensitivity to minority-balanced adjustment classes. Overall, the proposed Random Forest design is reproducible, interpretable, and flexible, as the same workflow can be adapted using alternative classifiers while preserving the aggregated data representation, statistical labeling framework, and evaluation protocol. This makes the methodology suitable not only for Dota 2 balance monitoring but also adaptable to broader hero-based balance analysis in other competitive multiplayer games. [1], [3].

2.7 Evaluation Protocol

To evaluate the classification performance of the proposed model, stratified 6-fold cross-validation is employed as the internal evaluation protocol. Stratified sampling preserves the original class proportions of OP, UP, and Balanced heroes in each fold, ensuring that minority classes remain represented during both training and validation. This strategy is particularly important for imbalanced multi-class problems, where random splitting can distort class proportions and yield unreliable performance estimates. By maintaining class consistency across folds, stratified cross-validation provides a more stable and representative evaluation of model behavior [18], [28].

In each fold, approximately five-sixths of the dataset is used for training, and the remaining one-sixth for validation. This process is repeated six times, allowing each subset to serve as validation data once. Model performance is then obtained by averaging the results across all folds. This iterative evaluation approach reduces reliance on a single data split and provides more stable and reliable estimates of model performance. Performance is evaluated using accuracy, precision, recall, and macro-averaged F1-score. While accuracy measures overall correctness, it can be misleading under imbalanced conditions where the majority class dominates. Therefore, this study emphasizes macro-F1 as the primary evaluation metric, as it assigns equal importance to all classes and better reflects the model's ability to detect minority classes (OP and UP), which are more relevant for balance recommendation tasks [18]. This is essential because correctly identifying minority adjustment classes (OP and UP) is more meaningful for balance recommendation than simply maximizing majority-class accuracy. To ensure the evaluation's reliability, performance metrics are reported as the mean across all folds, with standard deviation to capture variability in model performance. This provides a more robust assessment compared to a single train–test split and reduces sensitivity to data partitioning.

To further improve reliability, internal model performance is reported as the mean score across folds, and a confusion matrix is used to examine class-specific prediction patterns. This allows inspection of which hero-status classes are most frequently confused, providing deeper insight beyond aggregate metrics alone and supporting more transparent model interpretation [16], [26]. In addition to aggregate metrics, confusion matrices are used to analyse class-specific prediction patterns. This analysis provides insight into the distribution of classification errors across classes, allowing for a more comprehensive understanding of model behavior beyond overall performance metrics.

2.8 Robustness Analysis and Baseline Comparison

To strengthen methodological validity, additional robustness analyses were conducted beyond the primary internal evaluation. These analyses were designed to assess the risk of feature leakage, verify processing through to class synchronization, examine sensitivity to labeling, and compare the proposed model with a simple baseline strategy.

First, a feature-leakage assessment was conducted based on the failures of win_rate and pick_rate as model input features. This experiment was necessary because these two variables are used in the statistical labeling framework described in Section 2.5 to define the OP and UP classes. Including them directly as predictive features can introduce leakage, where the classifier learns variables that explicitly determine the target label rather than broader patterns of hero performance. After removing these two features, the Random Forest classifier still retained meaningful predictive ability, indicating that combat, economy, impact, and exposure features still provide useful signals for hero balance classification.

Second, class imbalance was explicitly addressed by applying a balanced class weighting scheme (*class_weight = balanced*) during Random Forest training. This weighting mechanism increases the contribution of minority classes (OP and UP) and prevents the model from being biased toward the majority Balanced class. As a result, the model is better able to learn the decision boundaries of the minority class, which are critical for identifying Buff and nerf candidates. This design prevented the classifier from collapsing into trivial majority-class prediction behavior and improved learning stability under imbalanced multi-class conditions [18], [28].

Third, sensitivity analysis was conducted on the threshold coefficient $k$used in the statistical labeling equations. As shown in Table 4, smaller values of $k$produced broader OP/UP assignments, while larger values generated increasingly conservative labeling dominated by the Balanced class. Although larger $k$values improved nominal internal accuracy, macro-F1 decreased due to reduced minority-class recognition. Among evaluated settings, $k = 0.3$provided the best trade-off between internal classification quality and external generalization, and was therefore selected as the final labeling parameter.

Finally, the proposed model was compared against a majority-class baseline that always predicts Balance for every hero. Across six patch transitions, this baseline achieved an average external accuracy of 0.606, reflecting the naturally dominant proportion of unchanged heroes in official patch notes. However, such a baseline completely failed to identify Buff and Nerf classes, making it unsuitable as a practical balancing recommendation strategy. In contrast, although the proposed model achieved a lower average external accuracy (0.559), it was able to actively identify minority adjustment candidates and generate actionable buff/nerf recommendations. A temporal-persistence baseline, in which predictions for patch t+1 follow actions from patch t, is not adopted due to structural inconsistencies across patches. Major updates may introduce gameplay changes, such as hero reworks, new hero additions, and modifications to core mechanics (e.g., facet systems), that significantly alter hero behavior and invalidate assumptions about temporal continuity. Therefore, such a baseline would not provide a fair or consistent reference for evaluation. In contrast, the proposed model leverages performance-driven signals to identify new balance candidates, providing more meaningful and actionable insights beyond simple heuristic strategies. Overall, these robustness analyses demonstrate that the proposed framework is not solely driven by labeling variables or class imbalance, but can capture meaningful performance patterns that generalize across patch transitions.

2.9 External Validation Across Patch Transition

To translate hero-status predictions into balancing recommendations, the predicted classes are mapped into patch actions: OP is interpreted as Nerf, UP is interpreted as Buff, and Balanced is interpreted as Balance. This mapping follows the balancing intuition that overperforming heroes are likely candidates for reductions, while underperforming heroes are candidates for targeted improvements. Such translation converts classification outputs into actionable balancing recommendations that are easier to interpret from a game-design perspective [24], [25]. For external validation, recommendations generated from patch $t$are compared against official hero adjustments introduced in patch $t + 1$. Official patch notes are manually summarized at the hero level into three categories: Buff, Nerf, and Balance. Heroes not explicitly mentioned in patch notes are categorized as Balance to ensure complete label coverage for all heroes in each transition. This design enables a full-patch comparison between model recommendations and actual developer-balancing decisions. Unlike conventional evaluation approaches that rely solely on internal validation, this protocol introduces a stricter, more realistic assessment by testing whether patterns learned from one patch generalize to subsequent patches under evolving game conditions. This patch-aware evaluation represents a key contribution of this study.

External validation is conducted across six consecutive patch transitions, namely 7.39-7.39b, 7.39b-7.39c, 7.39c-7.39d, 7.39d-7.39e, 7.39e-7.40, and 7.40-7.40b. This patch-to-patch evaluation is intentionally stricter than standard internal validation because it tests whether patterns learned from one metagame remain predictive under the next patch environment, where hero interactions, item changes, and systemic balancing updates may shift the competitive landscape [10]–[15]. Therefore, this evaluation better reflects real-world deployment conditions for balance recommendation systems. The selected patch range (7.39–7.40b) is chosen to maintain consistency in core gameplay mechanics. Major updates beyond this range introduce structural changes, such as modifications to hero systems (e.g., facet adjustments or removals), which significantly alter hero behavior and invalidate direct comparison across patches. Restricting the analysis to structurally consistent patches ensures that observed differences are driven primarily by balance adjustments rather than fundamental gameplay redesign. Overall, this external validation framework provides a more realistic benchmark for evaluating balance recommendation systems, as it reflects the dynamic and evolving nature of real-world game balancing processes.

3. Results and Discussion

This section presents the performance of the proposed model from both internal evaluation and external patch-to-patch validation, along with key interpretations. The model achieves strong internal performance but only moderate external accuracy (0.559), indicating a gap between the statistical patterns in professional match data and the actual

developer-balancing decisions. A consistent observation is that many heroes receiving Buff or Nerf actions are predicted as Balanced, suggesting that the model behaves conservatively and requires strong performance signals to classify extreme classes (OP/UP). This behavior is influenced by the threshold-based labeling scheme (μ ± kσ), class imbalance, and the use of aggregated performance features. As a result, the model is more effective at identifying clear outliers rather than subtle balance adjustments.

Comparison with a majority-class baseline (always predicting Balanced) shows that although the baseline achieves higher accuracy (0.606), it fails to detect Buff and Nerf classes. In contrast, the proposed model provides actionable predictions for minority classes, making it more useful as a decision-support tool despite lower overall accuracy. Furthermore, robustness experiments, including feature exclusion to assess potential data leakage and sensitivity analysis of the threshold parameter, indicate that the model does not rely solely on labeled variables and maintains meaningful predictive capability. Overall, the results suggest that professional match telemetry can provide useful early signals for balance monitoring, but cannot fully capture developer decisions, which are influenced by broader gameplay and design considerations. The following subsections present detailed evaluation results.

3.1 Stratified 6-Fold Cross-Validation

The proposed model achieves strong internal performance, with an accuracy of 0.94 and macro-F1 of 0.84, indicating effective classification across imbalanced classes. The use of stratified cross-validation ensures that class proportions are preserved in each fold, making the evaluation more reliable under imbalance conditions. The reported metrics are averaged across folds, with standard deviation included to reflect variability and ensure the reliability of the evaluation. The relatively low variation across folds indicates stable model performance under different data partitions. However, the confusion matrix (Table 6) shows that most misclassifications occur between the OP and Balanced classes. This suggests that some heroes exhibit partial dominance (e.g., high win rate but moderate pick rate), making them difficult to distinguish under the current threshold-based labeling scheme.

Table 6. Confusion Matrix (6-Fold Aggregate)

True / Pred	*Balance*	*OP*	*UP*
Balance	*92*	*0*	*0*
OP	*6*	*6*	*0*
UP	*1*	*0*	*21*

Analysis of the confusion matrix reveals that most predictions for the Balanced class are correct, while misclassifications primarily occur between the OP and Balanced classes. This suggests that some heroes exhibit partial dominance (e.g., high win rate but moderate pick rate), making them difficult to distinguish under the current threshold-based labeling scheme. Despite the high internal performance, these results are interpreted cautiously due to the potential influence of labeling variables (win_rate and pick_rate). However, the robustness analysis in Section 2.8 shows that the model retains meaningful predictive power even when these variables are excluded, indicating that performance is not solely driven by data leakage. Overall, the internal evaluation demonstrates that the model can effectively learn patterns of hero performance, while also highlighting limitations in distinguishing borderline cases. This behavior is consistent with the statistical labeling framework (μ ± kσ), where only strong deviations are classified as OP or UP. As a result, borderline cases tend to be categorized as Balanced, contributing to conservative prediction patterns

3.2 External Validation (Patch Transition)

External validation evaluates the model's ability to generalize across patch transitions by comparing predictions derived from patch t with actual developer actions implemented in patch t+1. This evaluation reflects real-world deployment conditions, where balancing decisions must be made based on historical data under evolving game dynamics. Unlike internal validation, which measures performance within a static dataset, external validation introduces temporal distribution shifts caused by patch updates. Therefore, this evaluation provides a more realistic and challenging benchmark for assessing model effectiveness.

We first construct a lightweight ground-truth table from official patch notes that records hero-level actions as Buff or Nerf (heroes not mentioned are treated as Balance). Next, model outputs are mapped to patch actions: OP to Nerf, UP to Buff, and Balanced to Balance. The predicted action table is then merged with the patch-note table using localized_name, producing a one-to-one comparison per hero. Across six consecutive patch transitions, external accuracy ranges from 0.436 to 0.672, with an average of 0.559. The best performance is observed in the 7.39b–7.39c transition (84/125; 0.672), while the lowest is observed in the 7.40–7.40b transition (55/126; 0.436). This variation indicates that model performance depends on the characteristics of each patch and the nature of balance updates. Finally, we evaluate agreement using accuracy and a confusion matrix, reported in the Results section (see Table 7 for the patch-to-patch summary and Table 8 for an example Confusion Matrix). A comparison with a majority-class baseline (always predicting Balanced) shows that the baseline achieves higher nominal accuracy (0.606) due to class

dominance. However, it fails to identify Buff and Nerf classes, which are critical for balance analysis. In contrast, the proposed model provides actionable predictions for these minority classes, making it a more suitable decision-support tool.

3.3 Patch-to-Patch External Validation Results

To evaluate real-world applicability, the proposed model is tested across six consecutive patch transitions by comparing predictions from patch *t* with official developer actions in patch *t+1*. The detailed results are presented in Table 7.

As shown in Table 7, external accuracy ranges from 0.436 to 0.672 across all transitions. The best performance is observed in the 7.39b - 7.39c transition (84/125; 0.672), while the lowest is in the 7.40 - 7.40b transition (55/126; 0.436). This variation indicates that model performance depends on patch characteristics and the nature of balance updates.

Table 5. External Accuracy (Patch-to-Patch vs Official Patch Notes)

Training Patch	Target Patch	Correct/Total	Accuracy
7.39	7.39b	68/125	0.5440
7.39b	7.39c	84/125	0.6720
7.39c	7.39d	73/125	0.5840
7.39d	7.39e	72/125	0.5760
7.39e	7.40	68/126	0.5397
7.40	7.40b	55/126	0.4365

A consistent pattern across all transitions is the dominance of correct predictions for the Balanced class, while many actual Buff and Nerf cases are misclassified as Balanced. This suggests that the model adopts a conservative prediction strategy, requiring strong statistical deviations before identifying adjustment candidates. The threshold-based labeling scheme and the inherent class imbalance in the Dataset influence this behavior. As a result, the model is more effective at identifying clear outliers than at subtle balance adjustments that may still trigger developer intervention. Furthermore, discrepancies between model predictions and actual developer actions can be attributed to factors beyond professional match statistics, including public matchmaking data, design objectives, hero role diversity, and systemic gameplay changes. These factors are not captured in the current feature set but play a significant role in balancing decisions. Overall, these results demonstrate that while the model cannot fully replicate developer decisions, it provides meaningful early signals for identifying potential balance issues. Therefore, the proposed approach should be interpreted as a decision-support tool rather than a replacement for developer judgment.

Such discrepancies are expected, as developer decisions are influenced by factors beyond professional match statistics, including public matchmaking data, design intent, role diversity, and broader systemic changes (e.g., item or gameplay mechanics updates).

Table 8. Confusion Matrix (Patch 7.39b with 7.39c)

True / Pred	*Balance*	*Nerf*	*Buff*
Balance	*62*	*5*	*12*
Nerf	*13*	*10*	*3*
Buff	*7*	*1*	*12*

An example confusion matrix for the 7.39b - 7.39c transition (Table 8) shows that most errors originate from Buff/Nerf cases predicted as Balanced, while correct Balance predictions remain dominant. This pattern is consistent across other patch transitions. Overall, these results confirm that while the model cannot fully replicate developer decisions, it can identify potential balance candidates and serve as a practical decision-support tool.

3.4 Sensitivity Analysis of Threshold Coefficient (k)

To evaluate the impact of the threshold coefficient (*k*) on labeling and model performance, a sensitivity analysis was conducted using multiple values (0.1–1.0). The results for the 7.39b-7.39c transition present in Table 9. Notably, this experiment excludes *win_rate* and *pick_rate* from the feature set to assess potential data leakage effects.

Table 6. Sensitivity Analysis of Threshold Coefficient (k) Without win_rate and pick_rate Features (Patch 7.39b → 7.39c)

K values	*Internal accuracy*	*macro f1 score*	*External accuracy*
0.1	*0.52*	*0.39*	*0.608 (76/125)*
0.2	*0.58*	*0.40*	*0.616 (77/125)*
0.3	*0.65*	*0.40*	*0.664 (83/125)*
0.5	*0.70*	*0.32*	*0.672 (84/125)*
0.7	*0.80*	*0.33*	*0.664 (83/125)*
1.0	*0.95*	*0.33*	*0.64 (80/125)*

As shown in Table 9, increasing *k* leads to a substantial improvement in internal accuracy (from 0.52 to 0.95), primarily due to the increasing dominance of the Balanced class. However, macro-F1 decreases (from 0.40 to 0.33), indicating reduced sensitivity in detecting minority classes (OP and UP). External validation performance remains relatively stable across different *k* values, with the highest accuracy at k = 0.5 (0.672), followed closely by k = 0.3 (0.664), suggesting that generalization across patch transitions is less sensitive to threshold variation than internal metrics. This suggests that external generalization is less sensitive to threshold variation compared to internal metrics. Although *k = 0.5* provides slightly higher external accuracy, it significantly reduces macro-F1, indicating weaker detection of minority classes. Therefore, *k = 0.3* is selected as the optimal configuration, as it provides a balanced trade-off between overall performance, sensitivity to balance-adjustment classes, and cross-patch generalization.

Importantly, the model maintains competitive performance even without *win_rate* and *pick_rate* features, indicating that predictions are not solely driven by the variables used in label construction. This result mitigates concerns regarding potential data leakage and confirms that other gameplay features contribute meaningful predictive signals. These findings highlight that threshold selection critically affects class distribution and model behavior, and should be chosen based on the intended objective, whether prioritizing overall accuracy or actionable detection of balance-adjustment candidates. These findings highlight that threshold selection is a critical design parameter in balance modeling, directly influencing both classification behavior and evaluation outcomes.

3.5 Extreme Hero Analysis

To complement the classification results, this study identifies extreme heroes as candidates for Buff and nerf based on deviations in win rate and pick rate. This approach provides an interpretable ranking of heroes who exhibit unusually strong or weak performance in professional matches.

Table 7. Top-3 Extreme Heroes

Type	*Hero*
Nerf	*Ember Spirit*
Nerf	*Ogre Magi*
Nerf	*Jakiro*
Buff	*Riki*
Buff	*Winter Wyvern*
Buff	*Visage*

As shown in Table 10 for nerf candidates, heroes such as Ember Spirit, Ogre Magi, and Jakiro consistently appear with high win rate and pick rate, indicating strong dominance and frequent selection. Conversely, heroes such as Riki, Winter Wyvern, and Visage are identified as buff candidates due to their low win and pick rates, suggesting weak performance and limited usage.

This analysis complements the classification model by highlighting specific heroes that may require balancing attention. However, similar to previous findings, these results should be interpreted as early signals rather than definitive balancing decisions, as they do not account for broader gameplay or design considerations.

4. Conclusion and Future Work

This study presentsa Random Forest–based methodfor classifying Dota 2 hero balance status using professional match telemetry and translating these classifications into balance recommendations. The approach integrates a transparent statistical labeling scheme and evaluates performance using both internal cross-validation and patch-to-

patch external validation. The results demonstrate strong internal performance, while external validation shows moderate agreement with developer patch decisions. This suggests that professional match data can provide useful signals for identifying possible balance concerns, but it cannot fully convey the intricacy of real-world balancing judgments.

Further study demonstrates that threshold selection, class imbalance, and the constraints of aggregated performance features all have an impact on model performance. Although simpler baselines may achieve higher nominal accuracy, they fail to identify actionable balance adjustments, highlighting the practical value of the proposed approach. First, model performance is sensitive to the threshold parameter (k), which affects class distribution and the trade-off between accuracy and minority-class detection. Second, robustness experiments confirm that the model maintains predictive capability even without win rate and pick rate features, reducing concerns about data leakage. Third, comparison with a majority-class baseline shows that, although simpler approaches may achieve higher accuracy, they fail to identify actionable Buff and nerf candidates.

Overall, the proposed framework should be interpreted as a decision-support tool that helps identify potential balance candidates, rather than a system that directly replicates developer decisions. Therefore, the proposed framework should be interpreted as a decision-support tool for identifying candidate heroes requiring further balancing evaluation, rather than a system intended to replace developer judgment.

Future work will focus on incorporating richer feature representations, including draft context, hero interactions, and temporal modeling, as well as exploring alternative labeling strategies to better align model outputs with real-world balancing practices.

Notation

WR : hero win rate
PR : hero pick rate
OP : overpower
UP : underpower
win_{high} : upper boundary for win rate
win_{low} : lower boundary for win rate
$pick_{high}$: upper boundary for pick rate
$pick_{low}$: lower boundary for pick rate
μ_{win} : mean (average) of win rate
σ_{win} : standard deviation of win rate
μ_{pick} : mean (average) of pick rate
σ_{pick} : standard deviation of pick rate
k = threshold scaling factor (k = 0.3)
$X \in \mathbb{R}^{h \times p}$: feature matrix (H = hero, p = fitur)
y: label vector (OP/UP/Balance)
$\hat{y}$: predicted labels
GPM = gold per minute
XPM = experience per minute

References

[1] C. Huang and S. D. Bruda, "Improved balance in multiplayer online battle arena games," *Acta Univ. Sapientiae, Inform.*, vol. 12, no. 2, pp. 183–204, Dec. 2020. https://doi.org/10.2478/ausi-2020-0011

[2] Y. Su, P. Backlund, and H. Engström, "Comprehensive review and classification of game analytics," *Serv. Oriented Comput. Appl.*, vol. 15, no. 2, pp. 141–156, Jun. 2021. https://doi.org/10.1007/s11761-020-00303-z

[3] X. Zhong and J. Xu, "Measuring the effect of game updates on player engagement: A cue from DOTA2," *Entertain. Comput.*, vol. 43, p. 100506, Aug. 2022. https://doi.org/10.1016/j.entcom.2022.100506

[4] S. García-Méndez and F. de Arriba-Pérez, "Explainable e-sports win prediction through Machine Learning classification in streaming," *Entertain. Comput.*, vol. 55, p. 101027, Sep. 2025. https://doi.org/10.1016/j.entcom.2025.101027

[5] J. Losada-Rodríguez, P. A. Castillo, A. Mora, and P. García-Sánchez, "The Explainability of Machine Learning Algorithms for Victory Prediction in the Video Game Dota 2," in *ITISE 2025*, Basel Switzerland: MDPI, Aug. 2025, p. 26. https://doi.org/10.3390/cmsf2025011026

[6] C. Zhao, H. Zhao, Y. Ge, R. Wu, and X. Shen, "Winning Tracker: A New Model for Real-time Winning Prediction in MOBA Games," in *Proceedings of the ACM Web Conference 2022*, New York, NY, USA: ACM, Apr. 2022, pp. 3387–3395. https://doi.org/10.1145/3485447.3512274

[7] Y. Peng, "The Application of Machine Learning in Predicting the Results of Popular eSports Games: Win Rate Prediction in MOBA and FPS Games," *Highlights Sci. Eng. Technol.*, vol. 85, pp. 1150–1156, Mar. 2024. https://doi.org/10.54097/dg0nm289

[8] S. Yangibaev, J. Mattiev, and S. Mokwena, "DotA 2 Match Outcome Prediction System Using Decision Tree Ensemble Algorithms," *Big Data Cogn. Comput.*, vol. 9, no. 12, p. 302, Nov. 2025. https://doi.org/10.3390/bdcc9120302

[9] V. P. K. Turlapati and M. R. Prusty, "Outlier-SMOTE: A refined oversampling technique for improved detection of COVID-19," *Intell. Med.*, vol. 3–4, p. 100023, Dec. 2020. https://doi.org/10.1016/j.ibmed.2020.100023

[10] M. Riess, "Automating model management: a survey on metaheuristics for concept-drift adaptation," *J. Data, Inf. Manag.*, vol. 4, no. 3–4, pp.

211–229, Dec. 2022. https://doi.org/10.1007/s42488-022-00075-5
[11] E. Yu, Y. Song, G. Zhang, and J. Lu, "Learn-to-adapt: Concept drift adaptation for hybrid multiple streams," *Neurocomputing*, vol. 496, pp. 121–130, Jul. 2022. https://doi.org/10.1016/j.neucom.2022.05.025
[12] D. M. V. Sato, S. C. De Freitas, J. P. Barddal, and E. E. Scalabrin, "A Survey on Concept Drift in Process Mining," *ACM Comput. Surv.*, vol. 54, no. 9, pp. 1–38, Dec. 2022. https://doi.org/10.1145/3472752
[13] A. L. Suárez-Cetrulo, D. Quintana, and A. Cervantes, "A survey on machine learning for recurring concept drifting data streams," *Expert Syst. Appl.*, vol. 213, p. 118934, Mar. 2023. https://doi.org/10.1016/j.eswa.2022.118934
[14] D. Lukats, O. Zielinski, A. Hahn, and F. Stahl, "A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams," *Int. J. Data Sci. Anal.*, vol. 19, no. 1, pp. 1–31, Jan. 2025. https://doi.org/10.1007/s41060-024-00620-y
[15] Q.-T. Tran, N.-A. Le-Khac, and M. Bertolotto, "Concept drift detection in image data stream: a survey on current literature, limitations and future directions," *Artif. Intell. Rev.*, vol. 59, no. 1, p. 33, Dec. 2025. https://doi.org/10.1007/s10462-025-11428-y
[16] R. R. Fernández, I. Martín de Diego, V. Aceña, A. Fernández-Isabel, and J. M. Moguerza, "Random forest explainability using counterfactual sets," *Inf. Fusion*, vol. 63, pp. 196–207, Nov. 2020. https://doi.org/10.1016/j.inffus.2020.07.001
[17] G. Hooker, L. Mentch, and S. Zhou, "Unrestricted permutation forces extrapolation: variable importance requires at least one more model, or there is no free variable importance," *Stat. Comput.*, vol. 31, no. 6, p. 82, Nov. 2021. https://doi.org/10.1007/s11222-021-10057-z
[18] K. Takahashi, K. Yamamoto, A. Kuchiba, and T. Koyama, "Confidence interval for micro-averaged F1 and macro-averaged F1 scores," *Appl. Intell.*, vol. 52, no. 5, pp. 4961–4972, Mar. 2022, doi: 10.1007/s10489-021-02635-5.
[19] D. R. Cano, "Statistics and stats of characters in video games," 2021.
[20] X. "Arcadia" Zhang and B. C. Keegan, "Characterizing disruptions in online gaming behavior following software patches," vol. 1, no. 1, pp. 1–21, 2022. https://doi.org/10.48550/arXiv.2207.02736
[21] C. Ringer *et al.*, "Time to Die 2: Improved in-game death prediction in Dota 2," *Mach. Learn. with Appl.*, vol. 12, no. November 2022, p. 100466, 2023. https://doi.org/10.1016/j.mlwa.2023.100466
[22] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable AI: A Review of Machine Learning Interpretability Methods," *Entropy*, vol. 23, no. 1, p. 18, Dec. 2020. https://doi.org/10.3390/e23010018
[23] S. Tymochko, E. Munch, J. Dunion, K. Corbosiero, and R. Torn, "Using persistent homology to quantify a diurnal cycle in hurricanes," *Pattern Recognit. Lett.*, vol. 133, pp. 137–143, May 2020. https://doi.org/10.1016/j.patrec.2020.02.022
[24] E. E. Cranmer, D.-I. D. Han, M. van Gisbergen, and T. Jung, "Esports matrix: Structuring the esports research agenda," *Comput. Human Behav.*, vol. 117, p. 106671, Apr. 2021. https://doi.org/10.1016/j.chb.2020.106671
[25] O. Sagi and L. Rokach, "Explainable decision forest: Transforming a decision forest into an interpretable tree," *Inf. Fusion*, vol. 61, pp. 124–138, Sep. 2020. https://doi.org/10.1016/j.inffus.2020.03.013
[26] L. Hakim, Z. Sari, A. R. Aristyo, and S. Pangestu, "Optimzing Android Program Malware Classification Using GridSearchCV Optimized Random Forest," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, May 2024. https://doi.org/10.22219/kinetik.v9i2.1944
[27] H. A. Rosyid, U. Pujianto, and M. R. Yudhistira, "Classification of Lexile Level Reading Load Using the K-Means Clustering and Random Forest Method," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, pp. 139–146, May 2020. https://doi.org/10.22219/kinetik.v5i2.897
[28] U. Krishnan and P. Sangar, "A Rebalancing Framework for Classification of Imbalanced Medical Appointment No-show Data," *J. Data Inf. Sci.*, vol. 6, no. 1, pp. 178–192, Feb. 2021. https://doi.org/10.2478/jdis-2021-0011