Counter Strike Performance Model (CSPM):

Predicting Win Percentage in CS2 Using a Power Law Model

By: Tyler Algigi @tyleralgigi
March 12, 2025

1. Introduction

Counter-Strike has stood the test of time with a simple yet effective premise. Originally released as a mod for Half-Life in 1999, its immense popularity led to Valve acquiring and publishing it as a standalone game in 2000. [1] Over the years, its core mechanics have remained largely unchanged, leading to its latest iteration: Counter-Strike 2 (CS2).

CS2 is a first-person shooter (FPS) where two teams of five players compete in a race to 13 round wins. While this is a basic overview, the nuances of high-level play are incredibly complex. At the professional level, teams spend thousands of hours refining their aim, movement, utility usage, and teamwork. The highest-ranked teams execute counter-strategies and in-game adjustments at a level that is almost impossible for an average player to comprehend.

Figure 1 Nertz (center) and Team Liquid post match in IEM Katowice 2025 Playin

With this complexity in mind, I wanted to take a different approach—rather than analyze every in-game factor, can we predict a CS2 team's win percentage using only round wins and losses? My goal was to develop a mathematical model based on historical match data that accurately estimates a team's win percentage. This model could be valuable for teams, analysts, organizations, and even betting markets looking to make more data-driven decisions.

2. Mathematical Approach

My inspiration for this model came from *Mathletics: How Gamblers, Managers, and Sports Enthusiasts Use Mathematics in Baseball, Basketball, and Football*. [2] The book discusses the Baseball Pythagorean Theorem, a simple yet effective model for predicting win percentage (Winston, 2006):

$$\frac{1}{1 + \left(\frac{RA}{RS}\right)^2} \tag{1}$$

Where:
RA = Runs Allowed
RS = Runs Scored

Figure 2 Simplest Form of the Pythagorean expectation aka Baseball Pythagorean Theorem

This model has remained accurate for decades, with newer variations improving its precision. [1] Many major sports have adapted their own versions of this theorem, leading me to wonder if Counter-Strike could have one as well. However, my initial attempts at adapting the Pythagorean formula to CS2 failed. The binary nature of CS2 rounds and the impact of in-game economy made the traditional model ill-suited for this context. After further research, I concluded that a Power Law-based equation was a better fit.

2.1. The Power Law

Instead of a linear model, I adopted a Power Law equation:

$$y = ax^b \tag{1}$$

Where:
y = The Dependent Variable
x = The Independent Variable
a, b = Constants

Figure 3 Simplest Form of the Power Law Equation

The reason this model is effective is that rounds won have a nonlinear impact on win percentage. This means that winning just a few extra rounds significantly increases a team's likelihood of winning the entire match. This formula was specifically optimized for CS2 and will be referred to as the Counter-Strike Performance Model (CSPM).

$$win\% = a \times \left(\frac{RW}{RW + RL}\right)^{b} \tag{2}$$

Where:
RW = Rounds Won
RL = Rounds Lost
a, b = Constants

Figure 4 Formula for Counter-Strike Performance Model (CSPM)

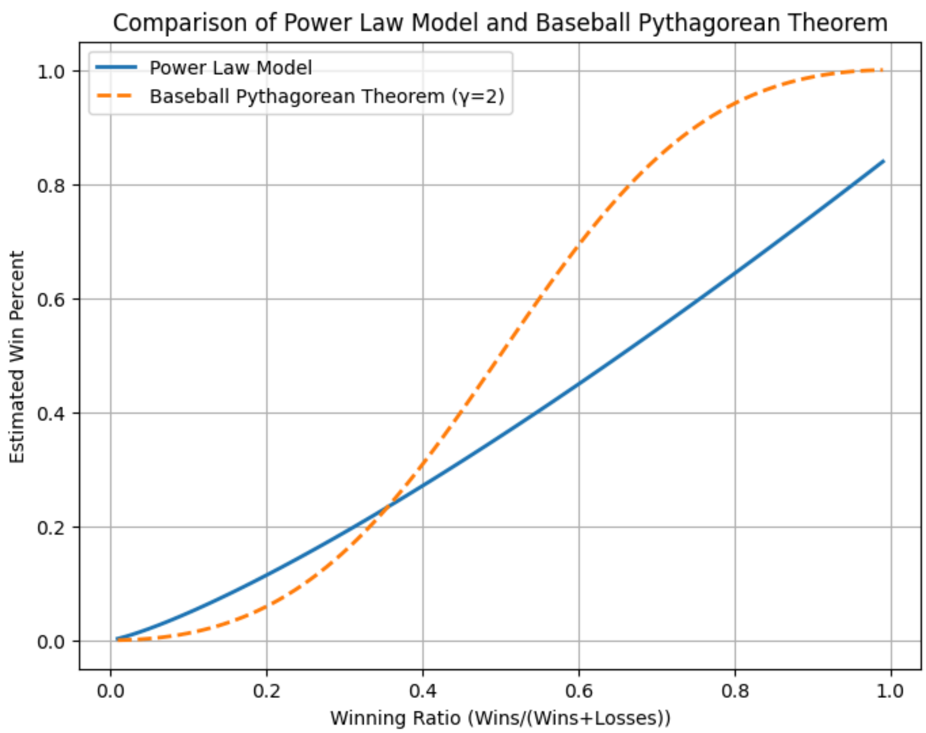


Figure 5 Comparison of Power Law Model and Pythagorean expectation

3. Data Collection

To ensure accurate and meaningful predictions, data was gathered from professional CS2 matches across tiers S, A, and B:

- S-Tier ⟶ Premier teams in the pro scene
- A-Tier ⟶ High-level teams below premier status
- B-Tier ⟶ Challengers, amateur, and semi-pro teams

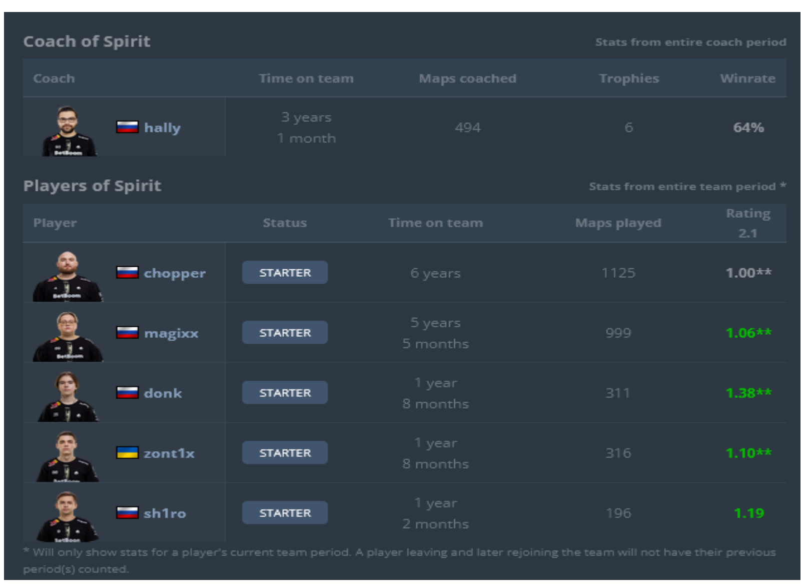
Two constraints were placed on the data:

1. **Tiers**: Lower-tier matches were excluded due to inconsistent data tracking.
2. **Minimum Match Number**: Only teams with at least 25 recorded matches were included to remove statistical noise from limited match data.

The final dataset included 147 teams, with the following features:

- Team Name
- Date of Most Recent Roster Change
- Total Rounds Played
- Total Rounds Won
- Total Games Played
- Total Games Won & Lost

For example, Team Spirit's current roster was finalized when sh1ro joined on December 17, 2023. [3] Thus, only matches after this date were included for that team.



Coach of Spirit — Stats from entire coach period

Coach	Time on team	Maps coached	Trophies	Winrate
hally	3 years 1 month	494	6	64%

Players of Spirit — Stats from entire team period *

Player	Status	Time on team	Maps played	Rating 2.1
chopper	STARTER	6 years	1125	1.00**
magixx	STARTER	5 years 5 months	999	1.06**
donk	STARTER	1 year 8 months	311	1.38**
zont1x	STARTER	1 year 8 months	316	1.10**
sh1ro	STARTER	1 year 2 months	196	1.19

* Will only show stats for a player's current team period. A player leaving and later rejoining the team will not have their previous period(s) counted.

Figure 6 Figure displays current roster of Team Spirit

4. Data Collection

To prepare the dataset, we first calculated a team's actual win percentage using the equation:

$$Actual\ Win\ Percentage = \frac{games\ won\ count}{games\ count}$$

Then, we applied the Power Law model to generate predicted win percentages and calculated the Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

y_i = Predicted Value

$\hat{y}_i$ = Actual Value

n = Total Number of oberservation

Figure 7 Simplest form of the Mean Squared Error formula

To evaluate model performance, we also used:

- F-Statistic & P-Value (ANOVA Test)
- Correlation (r) & Correlation P-Value (Pearson Test)

All data processing and calculations were performed in Python using pandas, NumPy, and SciPy. The optimal values produced from the calculation for a and b are 3.318 and 2.768 respectively.

```
[6]: from scipy.optimize import minimize
from scipy.stats import f_oneway, pearsonr
from sklearn.metrics import mean_squared_error

def power_law_eq(a, b, RW, RL):
    return a * (RW/(RW+RL))**b

# Define the cost function (e.g., mean squared error)
def cost_function(params, data):
    a, b = params
    predictions = a * (data['round_wins_count']/(data['round_wins_count']+data['round_lost_count']))**b
    return np.mean((predictions - data['win_percent']) ** 2)

# Initial guess for a and b
initial_guess = [0, 0]

# Optimization
result = minimize(cost_function, initial_guess, args=(team_data))

# Extract optimal values
a_optimal, b_optimal = result.x

print(result.x)

team_data['est_win_percent'] = team_data.apply(lambda x: power_law_eq(a_optimal, b_optimal, x['round_wins_c
team_data['difference'] = team_data['win_percent'] - team_data['est_win_percent']

print(team_data[['team_name', 'est_win_percent','win_percent', 'difference']])

mse = mean_squared_error(team_data['win_percent'], team_data['est_win_percent'])

print(mse)

f_statistics, p_value = f_oneway(team_data['win_percent'],team_data['est_win_percent'])
corr, p_value_corr = pearsonr(team_data['win_percent'],team_data['est_win_percent'])

print(f_statistics, p_value, corr, p_value_corr)
```

Figure 8 This figure displays the Python code used to calculate the most optimal a and b constants and the statistical evaluation metrics.

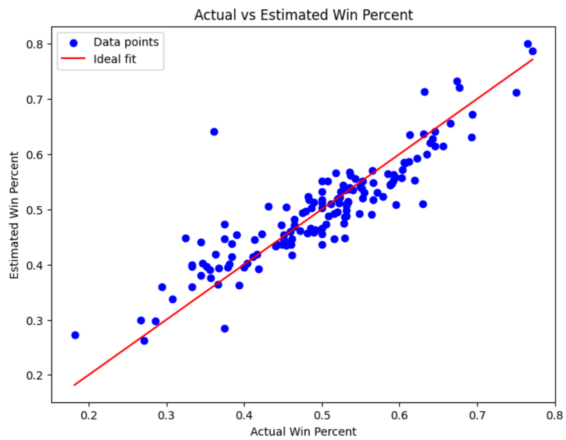


Figure 9 Figure displays a scatter plot with x axis representing Actual Win pct. and y axis representing Est. Win pct.

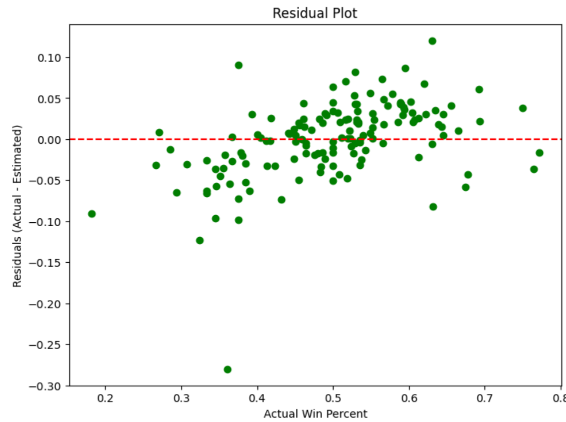


Figure 10 Figure is a scatter plot that displays the residuals (difference between actual and estimated win percentages)

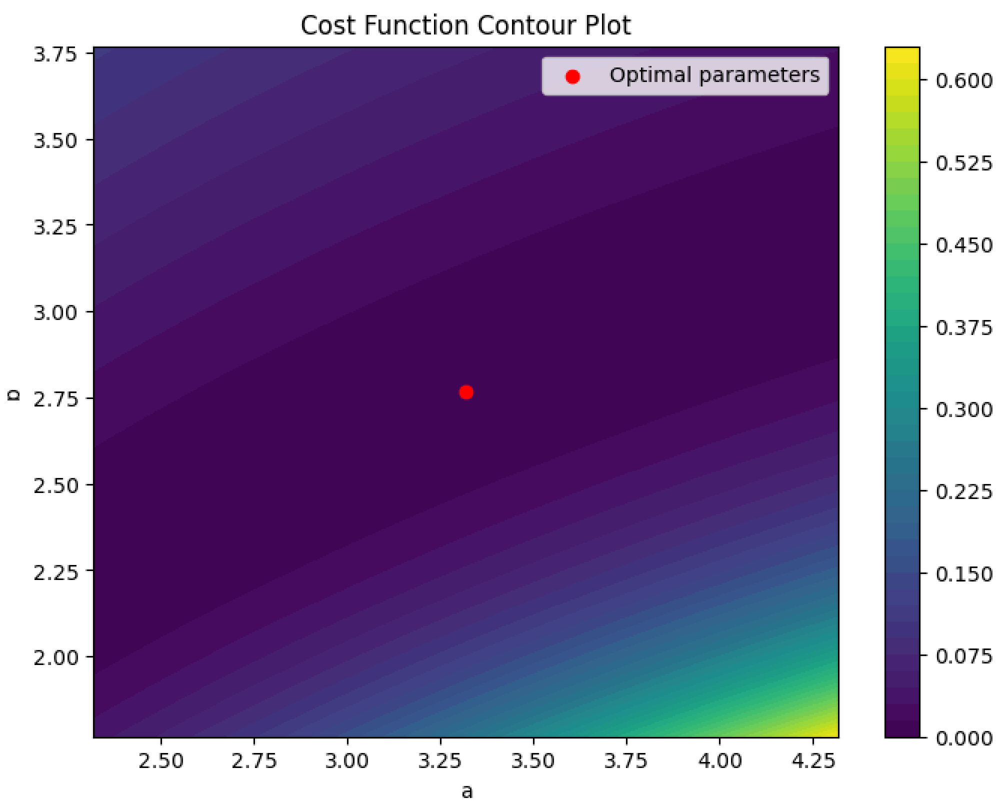


Fig 11 Figure is a contour plot displays the values of a on the X axis and b on the Y axis with bands of colors representing the values of MSE on the Z axis.

5. Model Evaluation

Our model achieved the following statistical results:

- Mean Square Error (MSE): 2.17×10^{-3}
- F-Statistic: 5.39×10^{-4}
- P-Value: 0.981
- Correlation (r): 0.899
- Correlation P-Value: 9.85×10^{-54}

Our evaluation results demonstrate that the Power Law model effectively predicts CS2 win percentage with high accuracy and statistical significance. The Mean Squared Error (MSE) of 2.17×10^{-3} indicates that, on average, the squared differences between the predicted and actual win percentages are very small, confirming that our model makes minimal errors. The F-statistic (5.39×10^{-4}) and p-value (0.981) suggest that there is no statistically significant difference between actual and predicted values, meaning our model is unbiased and does not systematically over- or under-predict win percentages. Furthermore, the correlation coefficient (r = 0.899) shows a strong positive relationship between the actual and estimated values, meaning that as actual win percentages increase, so do our predictions in a nearly linear fashion. Finally, the extremely small correlation p-value (9.85×10^{-54}) confirms that this correlation is highly statistically significant and not due to random chance. Since the model is well-fitted and unbiased, it can be used for forecasting future match results. Overall, these results validate our model as a robust and reliable predictor of team performance in CS2.

6. Applications & Future Work

With CSPM demonstrating strong statistical performance, it has several practical applications. Similar to the Baseball Pythagorean Theorem, this model serves as a predictive formula for estimating a team's expected win percentage. By looking beyond a team's raw win-loss record, CSPM accounts for round performance, which can be a more accurate measure of team strength. Traditional win-loss records are often influenced by luck, external factors, and close matches, making CSPM a valuable tool for identifying whether a team is over performing or underperforming relative to expectations. Additionally, this model can be used to calculate how many matches a team should have won based on round differential, offering deeper insights into team consistency and performance trends. As a result, CSPM can be applied in various areas, including team analysis, organizational decision-making, and betting/gambling markets. Players and analysts can utilize CSPM to evaluate expected match win rates, helping them make data-driven strategic decisions.

6.1. Further Improvements

Looking ahead, CSPM has room for further refinement and expansion. Just as the Baseball Pythagorean Theorem evolved over time, new formulas and modifications can enhance CSPM's accuracy, predictiveness, and overall usefulness. Several factors could be incorporated to improve its precision, such as economy data, map-specific models, and player-level contributions. Economy data would allow CSPM to adjust for the impact of pistol rounds, buy rounds, and economic advantages, while map-based models could account for varying win rates across different maps. Additionally, factoring in individual player contributions would provide a more granular approach to team performance evaluation. By implementing these improvements, CSPM could become an even more robust and reliable tool for predicting team success in Counter-Strike 2.

7. Conclusion

The Counter-Strike Performance Model (CSPM) effectively predicts team win percentages in CS2 using a Power Law equation. Our results confirm that CSPM provides a highly accurate, statistically significant method for estimating match outcomes based on round performance.

While CSPM already provides valuable insights, further improvements—such as incorporating economy data and map-based factors—could enhance its predictive power even further. As esports analytics continue to evolve, models like CSPM will play a crucial role in understanding performance, optimizing strategies, and making data-backed decisions in Counter-Strike.

References

[1] **Wikipedia Article (Counter-Strike):**
Wikipedia contributors. (n.d.). *Counter-Strike*. In *Wikipedia, The Free Encyclopedia*. Retrieved March 12, 2025, from https://en.wikipedia.org/wiki/Counter-Strike

[2] **Book Reference:**
Winston, W. L. (2006). *Mathletics: How gamblers, managers, and sports enthusiasts use mathematics in baseball, basketball, and football*. Princeton University Press.

[3] **HLTV Page (SPIRIT):**
SPIRIT. (n.d.). *HLTV*. Retrieved March 12, 2025, from https://www.hltv.org/team/7020/spirit#tab-rosterBox