Model Training Summary

1. Score Prediction Models

Feature Set

win_numeric

score_against

first_downs

third_down_comp

yards

pass_yards

rush_att

rush_yards

redzone_comp

redzone_att

sacks_num

turnovers_forced

turnover_diff_pct

possession_time

redzone_efficiency

third_down_efficiency

yards_per_play

pass_completion_pct

Why Linear Regression Was Selected

Model	RMSE	R^2
Linear Regression	**4.349**	**0.8048**
Random Forest	4.367	0.8031
XGBoost	4.358	0.8039

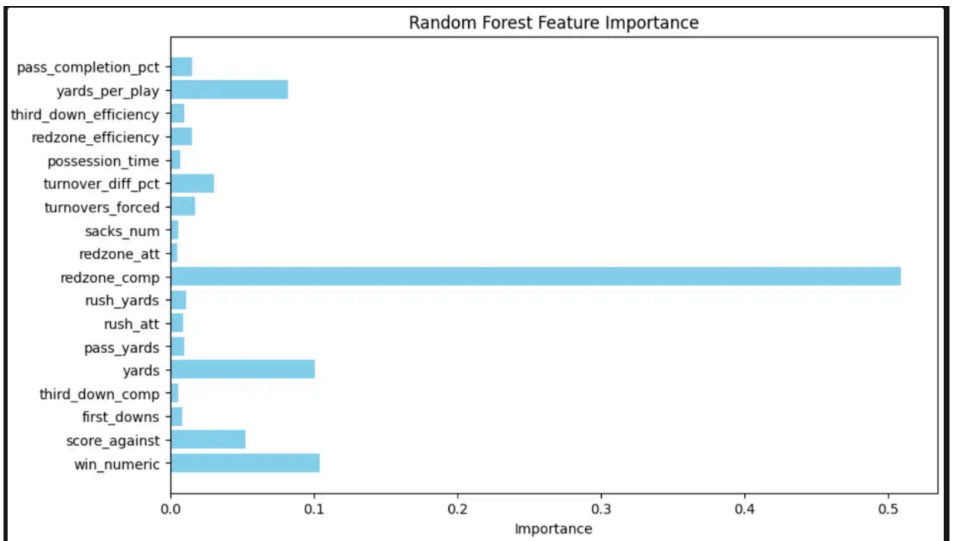
All three models were close, but Linear Regression delivered the **best RMSE and R^2 with the most stable error pattern**. Since the model is both simpler and more accurate, it was chosen to be the prediction model.

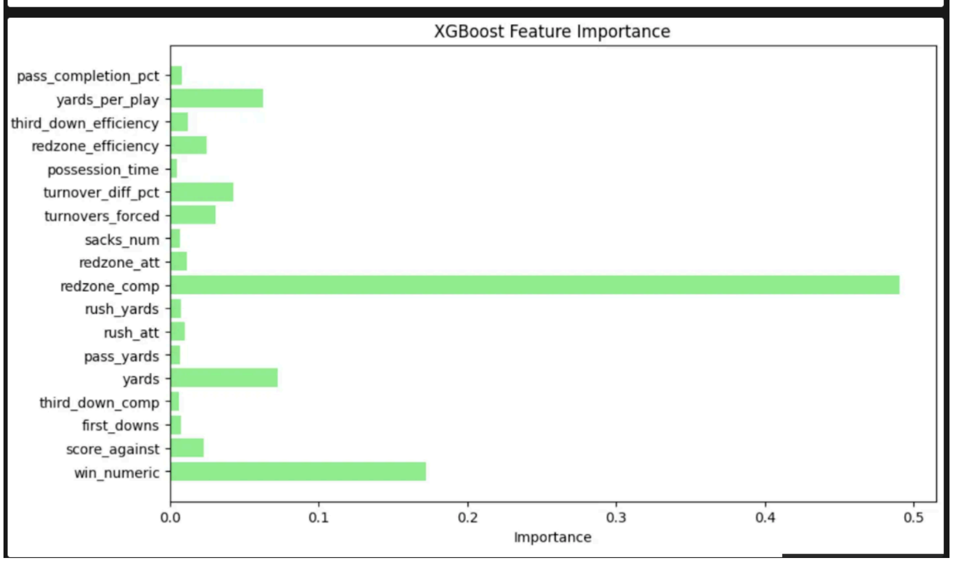
Importance Plots (Random Forest & XGBoost)

Random Forest & XGBoost importances clarify which features drive scoring:

- **redzone_comp** dominates which shows that finishing drives matters.
- **win_numeric** is also a driving feature, which shows that teams score more points when they win.
- **yards_per_play, yards, score_against,** and **turnover_diff_pct** carry strong weight.

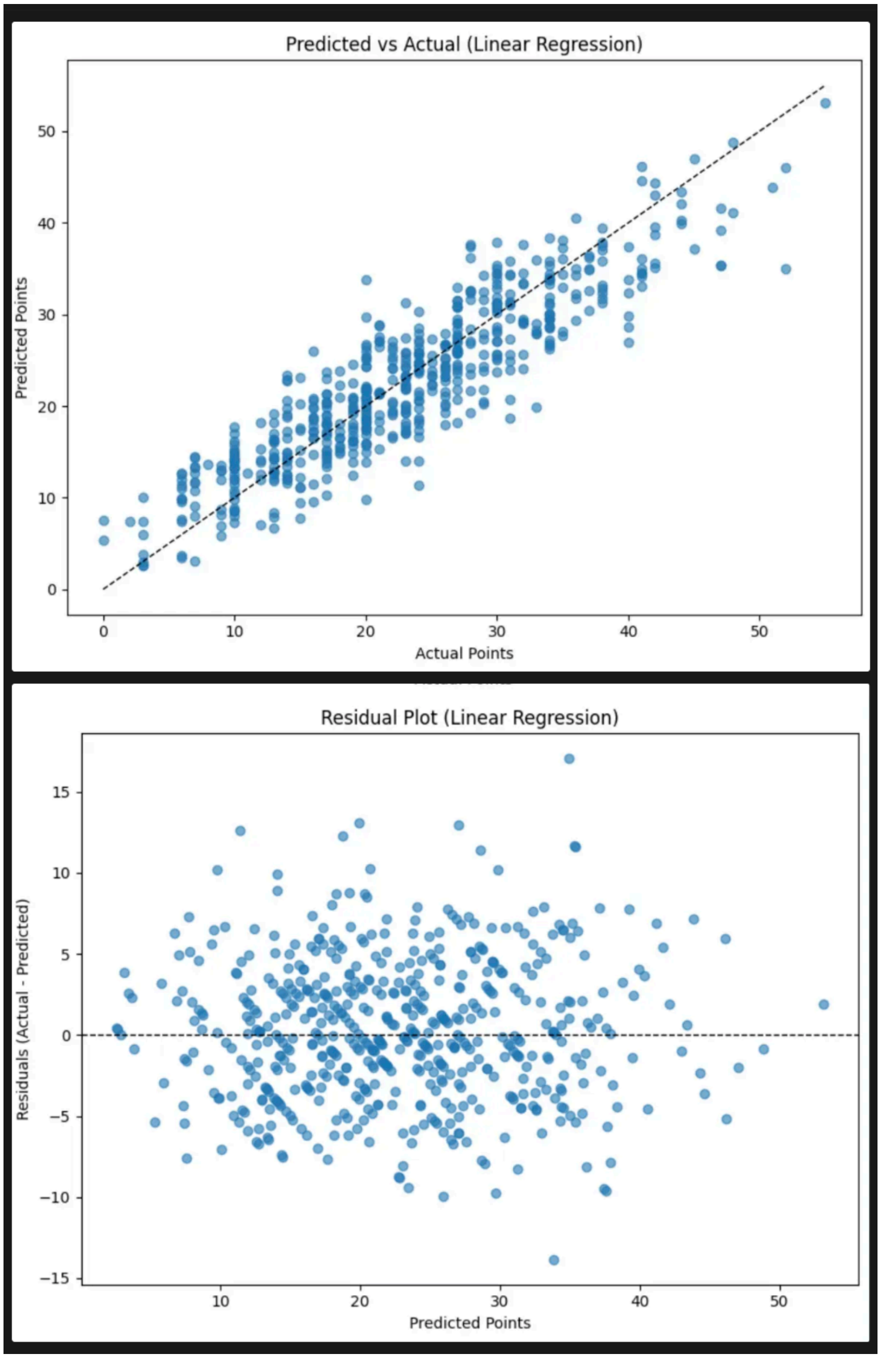
The distribution matches realistic football behavior: **high scoring games + yards + turnover & redzone efficiency = points.**





Charts

- **Predicted vs Actual:** A clean diagonal pattern with predictable spread at high outliers.
- **Residual Plot:** Noise centered around zero with no structural bias.



2. Win/Loss Prediction Model (Logistic Regression)

Feature Set

turnovers_forced

turnover_diff_pct

redzone_comp

redzone_att

possession_time_minutes

pass_att

rush_att

location

redzone_efficiency

turnover_impact

redzone_efficiency = redzone_comp / redzone_att

- Tracks how well a team finishes drives

turnover_impact = turnover_diff_pct * possession_time_minutes

- Captures how much turnovers actually mattered by combining turnover margin with possession time

Why Logistic Regression Was Selected

Model	Accuracy	F1 Score	Brier Score	ROC-AUC
Logistic Regression	**0.798**	**0.799**	**0.1386**	**0.8837**
XGBoost	0.767	0.773	0.1637	0.8535

Logistic Regression wins across the board. But the deciding factor wasn't just **accuracy**, it was **calibration**.

XGBoost Overconfidence Issue

XGBoost frequently produced **extreme probability outputs** (e.g., **0.99 vs 0.01**) even when games were far closer. This leads to:

- Overconfident predictions
- Worse Brier Scores

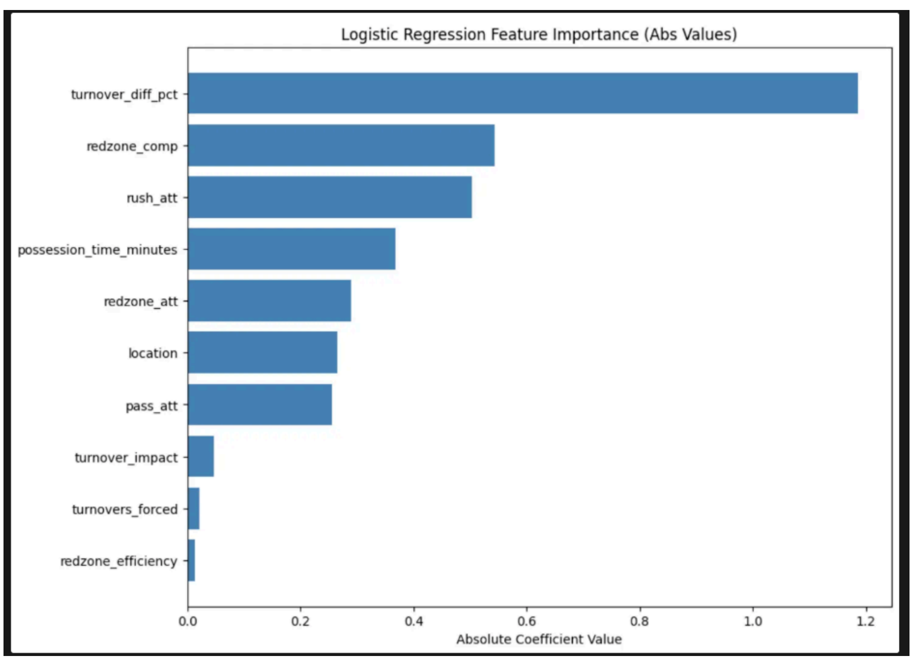
Logistic Regression produced **smoother probabilities**, which was important to have for the meta-model integration.

Importance Plot (Logistic Regression)

Absolute coefficients highlight:

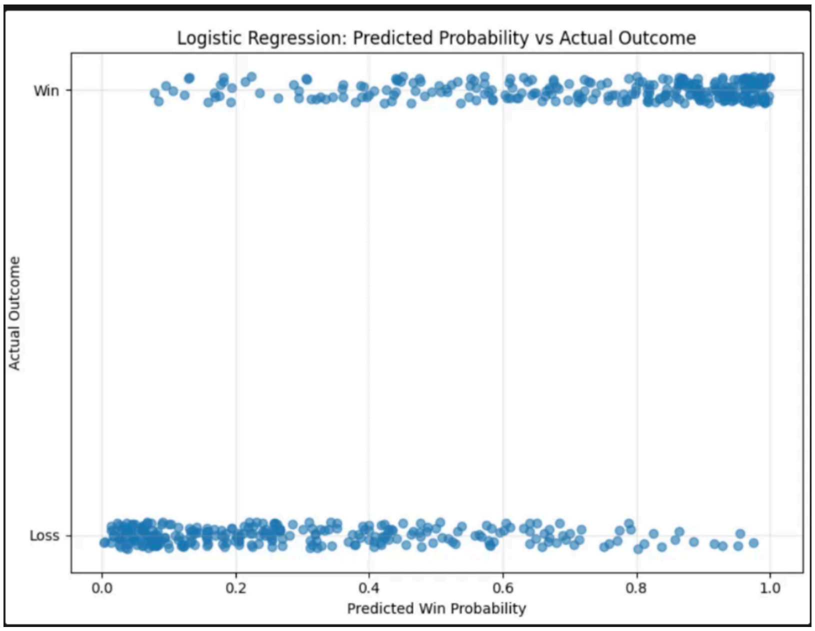
- **turnover_diff_pct** as the strongest driver of winning.
- **redzone_comp**, **rush_att**, and **possession_time_minutes, pass_att** as central factors.
  location (home/away) plays a meaningful role too (while was not shown on heatmap) and makes sense based on real world scenarios

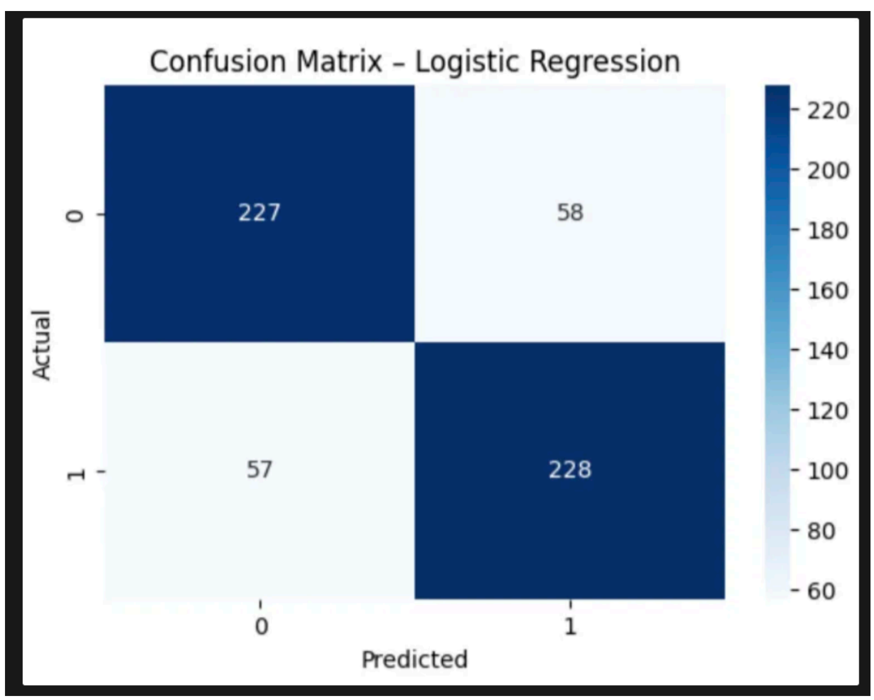
Again, the ordering reflects actual game dynamics: **finish drives, keep the ball, and win the turnover battle.**



Charts

- **Predicted Probability vs Outcome:** Clear separation between wins and losses; no extreme compression.
- **Confusion Matrix:** Balanced true positive & true negative distribution and controlled error rates.





3. Meta Model (Score + Win/Loss Integration w/ Vegas-Style Adjustment)

Why Build a Meta Model?

The standalone models occasionally disagreed. For example:

- A team that was projected to **win**, **scored less** than the team it was against
- Even when not conflicting, a team received **0.90 win probability**, but the score model predicted a **close margin**.

Since **predictions need to reinforce each other, not contradict**, Vegas lines were added to **blend** spread, total, and win probability.

Vegas-Style Adjustment

A light correction aligns the two:

```
def vegas_meta(score_pred, win_prob):
    bias = (win_prob - 0.5) * 2   # ranges from -1 to +1
    return float(np.clip(score_pred + bias, 0, None))
```

The ±1 point bias keeps the score aligned with the win probability without distorting the regression output.

Meta Model Performance

Metric	Value
RMSE	**4.403**
R^2	**0.800**
Accuracy	**0.782**

F1 Score	**0.777**
Brier Score	**0.157**
ROC-AUC	**0.877**

The score metrics **dip by very little** compared to the raw models since the Vegas model adds a small calibration bias which forces the scores and win probabilities to agree.

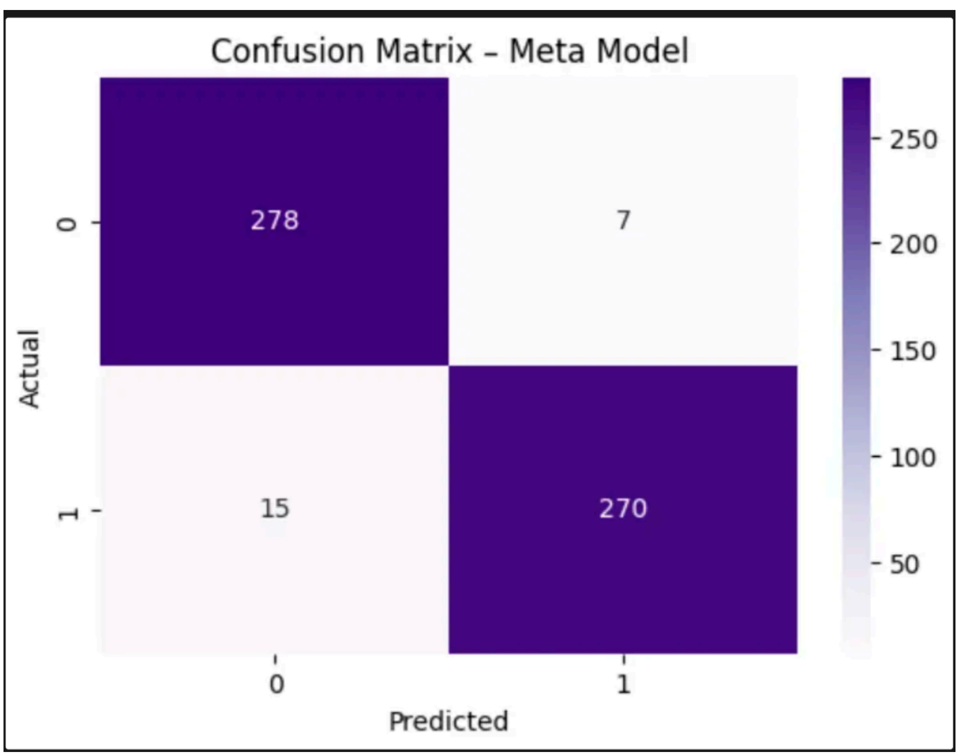
- However, there was one behavior that drastically improved.

Meta Confusion Matrix

Compared to the standalone logistic classifier:

- **Logistic**: false negative and false positive around **57 and 58**
  Meta: false negative and false positive drop to **15 and 7**

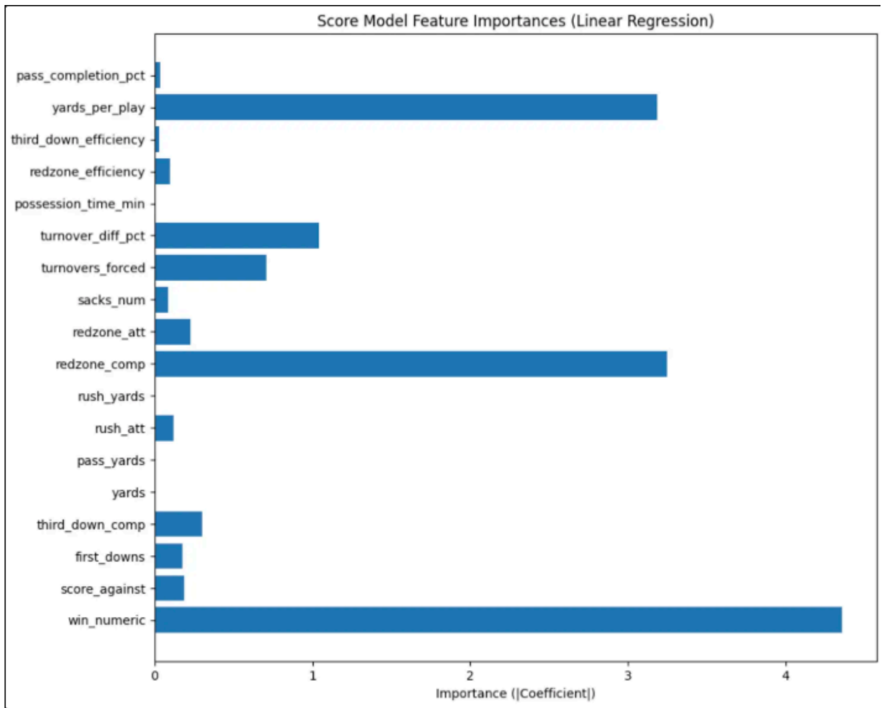
The score signal helps clean up borderline predictions, producing a **more decisive and internally consistent model which is exactly what I was going for!**

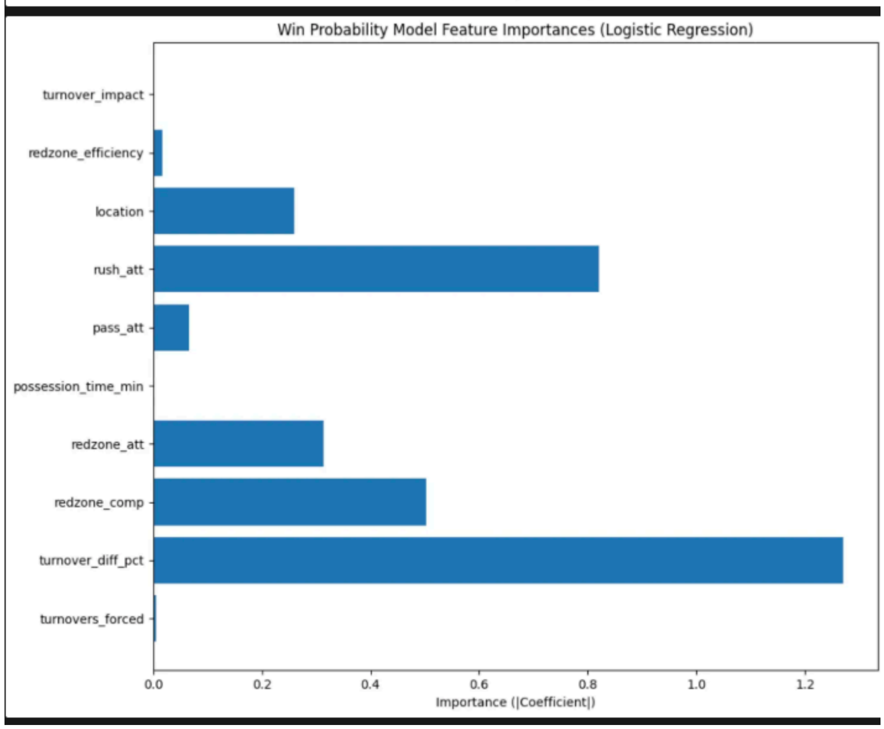


Importance Plots

- Score-side meta importances highlight **win_numeric**, **redzone_comp**, **yards_per_play**, and **turnover_diff_pct** as major scoring drivers.
- Win-side importances more-or-less mirror the earlier logistic model: turnovers, redzone usage, rushing volume, and home/away factors dominate.

The meta model behaves like a **unified prediction engine** rather than two disconnected estimators.





Charts

- Meta model **points hug the diagonal harder** and the variance shrinks.
- Meta **probabilities are calibrated by score margin**, so they spread cleanly from ~0.3 to ~0.9 with far fewer extremes.