Formula 1 Overtake Prediction

Analisi Predittiva con Machine Learning su Dati di Telemetria

Relazione Tecnica di Progetto

Autori:

Angelo Fusco & Mattia Fanzini

Università:

Università degli Studi di Salerno

Data:

14 Gennaio 2026

Indice

1 Introduzione

La Formula 1 moderna è definita dai dati. Ogni decisione strategica, dal pit stop alla gestione gomme, si basa su modelli predittivi. Questo progetto risponde alla domanda: *"È possibile prevedere la probabilità di un sorpasso utilizzando esclusivamente dati telemetrici storici?"*.

Utilizzando dati estratti dalla libreria `FastF1`, abbiamo sviluppato una pipeline di Machine Learning completa. Sono stati confrontati tre algoritmi (Logistic Regression, Random Forest, XGBoost) per classificare l'evento "Sorpasso". Il modello migliore è stato infine integrato in una Dashboard interattiva sviluppata con `Streamlit` che simula un muretto box reale durante il Gran Premio di Monza.

2 Data Engineering e Preprocessing

2.1 Costruzione delle Feature Relative

I dati grezzi di telemetria (velocità, giri motore) non sono predittivi se presi isolatamente. È stato sviluppato il modulo `relative_feature_builder.py` per calcolare metriche "differenziali" che catturano la dinamica del duello tra due piloti. Le feature principali ingegnerizzate includono:

- **Delta LapTime:** Differenza temporale sul giro tra attaccante e difensore.
- **Delta Tyre Life:** Differenza di usura degli pneumatici (in giri).
- **Compound Advantage:** Vantaggio di mescola codificato numericamente (es. Soft=3 vs Hard=1).
- **Estimated Gap:** Stima del distacco fisico basata sul delta tempo.

2.2 Pulizia e Outlier Detection

Per garantire la qualità del training, è stato necessario filtrare eventi non competitivi come Pit Stop o regime di Safety Car. Invece di usare soglie arbitrarie, abbiamo implementato nel modulo `relative_feature_builder.py` un filtro statistico basato sulla distribuzione dei tempi:

$$T_{limit} = \mu_{session} + 2\sigma_{session}$$

I giri con tempo superiore a questa soglia sono stati rimossi. Inoltre, i valori mancanti sono stati gestiti tramite riempimento a zero (`fillna(0)`) per mantenere la consistenza del dataset.

2.3 Bilanciamento delle Classi (SMOTE)

L'analisi del dataset ha rivelato un forte sbilanciamento: i sorpassi rappresentano una minoranza degli eventi rispetto ai giri di inseguimento. Per evitare che il modello favorisse la classe maggioritaria ("Nessun Sorpasso"), è stata applicata la tecnica **SMOTE** (Synthetic Minority Over-sampling Technique) nel modulo `feature_processor.py`, impostando i vicini ($k\_neighbors$) dinamicamente in base alla numerosità della classe minoritaria (minimo 5).

3 Analisi dei Modelli e Risultati

La fase di training (`model_trainer.py`) ha confrontato tre classificatori utilizzando uno split stratificato 80/20. Di seguito i risultati esatti ottenuti dai report di validazione.

3.1 Logistic Regression

Impostata con pesi di classe bilanciati (`class_weight='balanced'`), ha mostrato una buona capacità di recupero (Recall) ma una precisione molto bassa.

- **Accuratezza:** 71.2%
- **Falsi Positivi:** 108
- **Note:** Il modello tende a sovrastimare la probabilità di sorpasso, generando troppi falsi allarmi.

3.2 Random Forest

Configurato con 100 estimatori, ha migliorato significativamente l'accuratezza globale.

- **Accuratezza:** 80.0%
- **Falsi Positivi:** 47

3.3 XGBoost (Modello Selezionato)

XGBoost si è rivelato il modello più robusto per l'utilizzo in strategia di gara. Sebbene il Random Forest abbia performance simili, XGBoost ha minimizzato il numero di Falsi Positivi (solo 38), risultando il più affidabile per evitare chiamate ai box errate basate su "sorpassi fantasma".

Tabella 1: Confronto Metriche (Dati da training_report.json)

Modello	Accuratezza	Falsi Positivi	ROC-AUC
Logistic Regression	71.2%	108	0.750
Random Forest	80.0%	47	0.778
XGBoost	**81.4%**	**38**	**0.757**

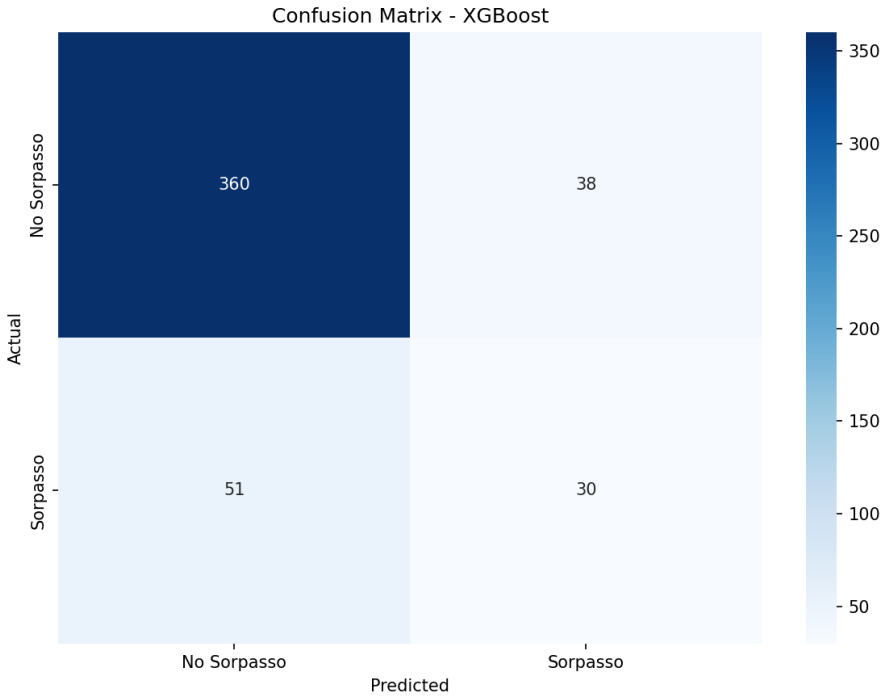

Figura 1: Matrice di Confusione - XGBoost (Accuratezza 81.4%)

4 Applicazione: F1 Strategy Room

Il culmine del progetto è l'applicazione web `app.py`, che carica il modello addestrato (`best_model.pkl`) e lo scaler (`scaler.pkl`) per effettuare previsioni in tempo reale.

Le funzionalità principali includono:

- **Simulazione Scenario:** L'utente può definire lo stato delle gomme (es. usura 12 giri vs 25 giri) e la mescola per calcolare la probabilità immediata di sorpasso.
- **Modalità Attacco/Difesa:** Un toggle permette di invertire i ruoli tra il pilota guidato e l'avversario.
- **Visualizzazione Tracciato:** Una mappa SVG interattiva del circuito di Monza mostra le posizioni stimate dei piloti.

5 Conclusioni

Il progetto ha dimostrato che l'uso di feature differenziali (Delta LapTime, Delta Tyre Life) permette di prevedere i sorpassi con un'accuratezza superiore all'81% utilizzando XGBoost. L'integrazione di tecniche come SMOTE e la rigorosa gestione degli outlier sono stati passaggi critici per ottenere risultati utilizzabili in uno scenario realistico di strategia di gara.