NFL Big Data Bowl 2025: Play Type Classifier

Abe Raouh and Benjamin Castle

CISC 5790 - Dr. Zhao Yijun

https://github.com/ibraouh/nfl-playtype-classifier

Introduction

- NFL Big Data Bowl 2025 dataset
- Apply machine learning to predict football play outcomes (Run vs Pass)
- Use player movement and other features
- Evaluate classification models and results

Football Concepts

- Pre-snap phase (40 seconds): Player movement before the ball snap
- Key positions: Wide Receiver (WR) and Running Back (RB)
- Focus: Predicting play outcome (Run or Pass)
- Importance of classifying plays

Data Exploration and Preprocessing

- Dataset Overview:
  - games.csv: Game details
  - plays.csv: Play-by-play data
  - players.csv: Player information
  - tracking_week_#.csv: Player movements
- Steps:
  - Clean missing values (5 rows dropped)
  - Filter important columns (quarter, down, yardsToGo)
  - Handle class imbalance (Run:Pass = 1:1.5)

Feature Engineering

- Extracted relevant columns (players.csv, games.csv, plays.csv)
- Filtered tracking data: BEFORE_SNAP + possession team
- Engineered Features:
- Total distance traveled by possession team
- Distance traveled by position (WR, RB, QB, etc.)

```python
# Filter rows where the possession team matches the club and frameType is BEFORE_SNAP
tracking_with_position = tracking_with_position[
    (tracking_with_position["club"] == tracking_with_position["possessionTeam"])
    & (tracking_with_position["frameType"] == "BEFORE_SNAP")
]
```

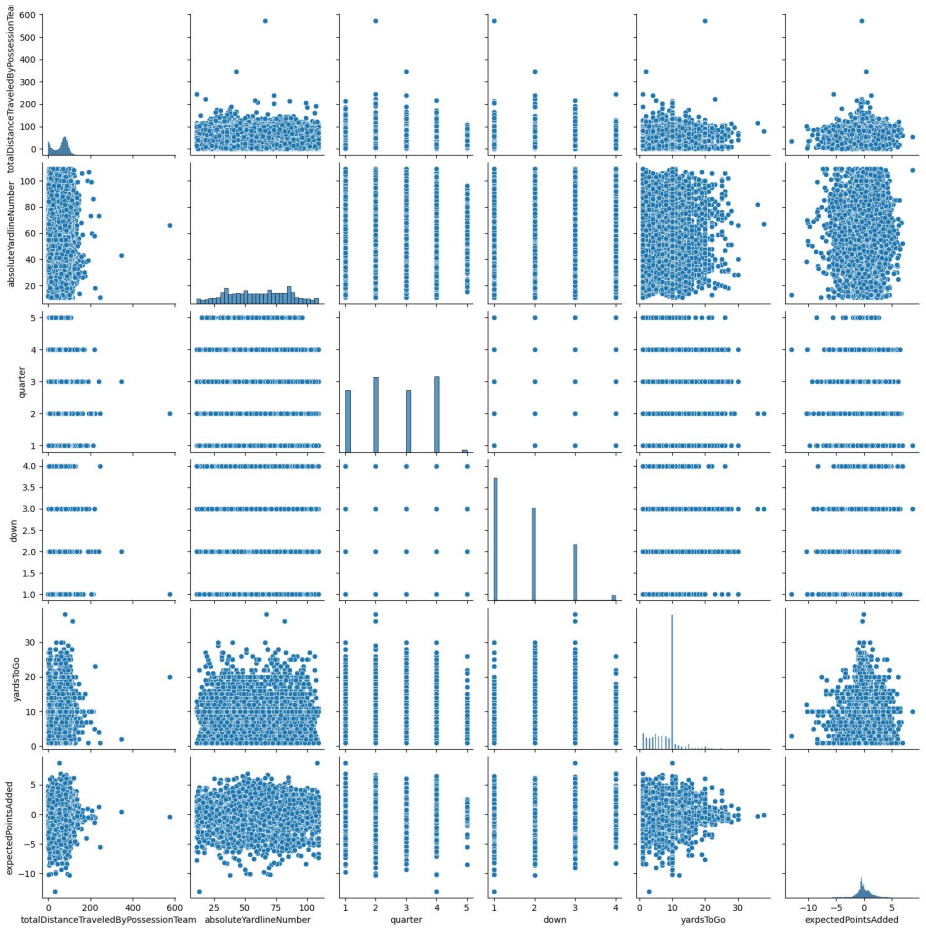
```python
# Calculate total distance traveled by possession team during a play
total_distance_possession_team = (
    tracking_with_position.groupby(["gameId", "playId"])["dis"]
    .sum()
    .reset_index()
    .rename(columns={"dis": "totalDistanceTraveledByPossessionTeam"})
)

# Calculate distances traveled for each position during a play
distance_by_position = (
    tracking_with_position.groupby(["gameId", "playId", "position"])["dis"]
    .sum()
    .reset_index()
)
```

```python
distance_by_position = (
    tracking_with_position.groupby(["gameId", "playId", "position"])["dis"]
    .sum()
    .reset_index()
)

# Pivot the data to get distances for each position as columns
distance_by_position_pivot = distance_by_position.pivot(
    index=["gameId", "playId"], columns="position", values="dis"
).fillna(0)

# Rename columns for clarity
distance_by_position_pivot.columns = [
    f"distance_{pos}" for pos in distance_by_position_pivot.columns
]
distance_by_position_pivot = distance_by_position_pivot.reset_index()

# Merge position distances and total possession distance into the main data
combined_data_with_distance = pd.merge(
    merged_data_sorted, distance_by_position_pivot, on=["gameId", "playId"], how="left"
)
combined_data_with_distance = pd.merge(
    combined_data_with_distance,
    total_distance_possession_team,
    on=["gameId", "playId"],
    how="left",
)
```

Model Building

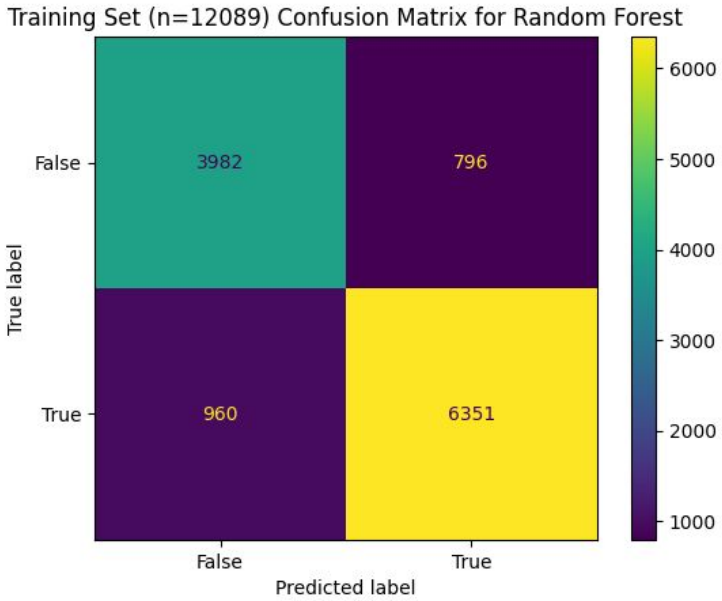
- LinearRegression attempted (no pairwise trends)
- RandomForestClassifier (100 trees, depth=12)
- TunedThresholdClassifierCV (Weighted F1)
- isDropback (indicator): Run=False, Pass=True

Linear Regression Struggles

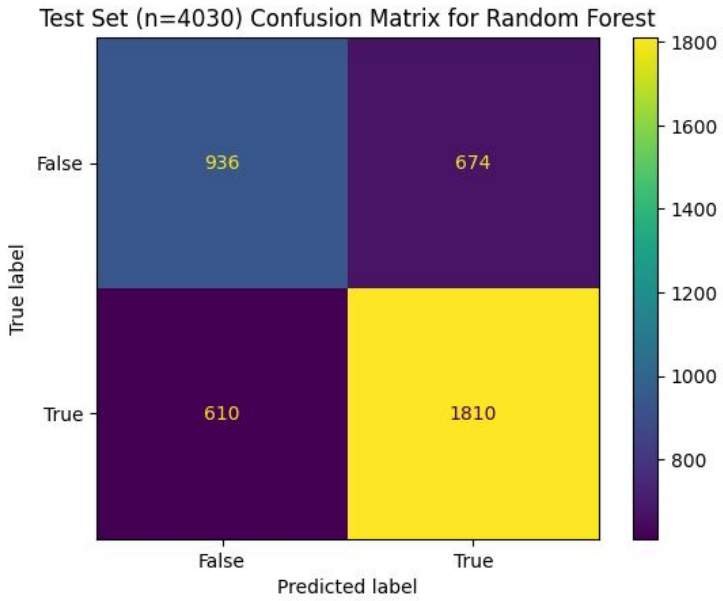


Random Forest

Accuracy: 85%
Precision: 86%

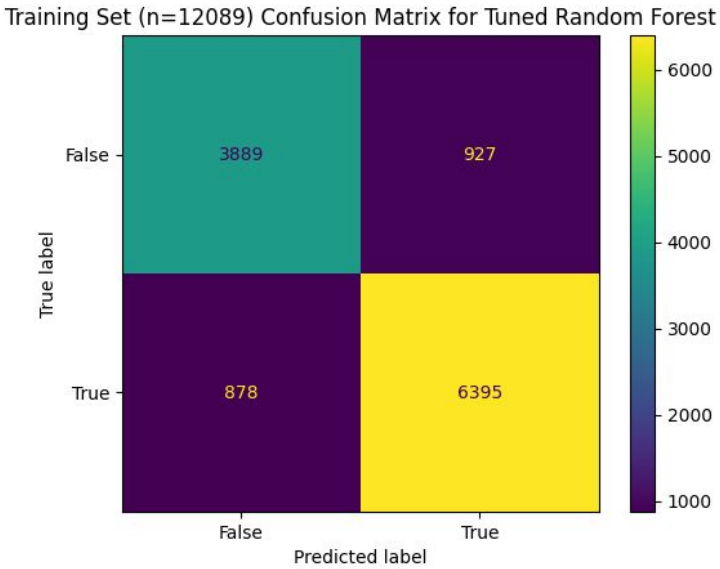


Accuracy: 68%
Precision: 68%

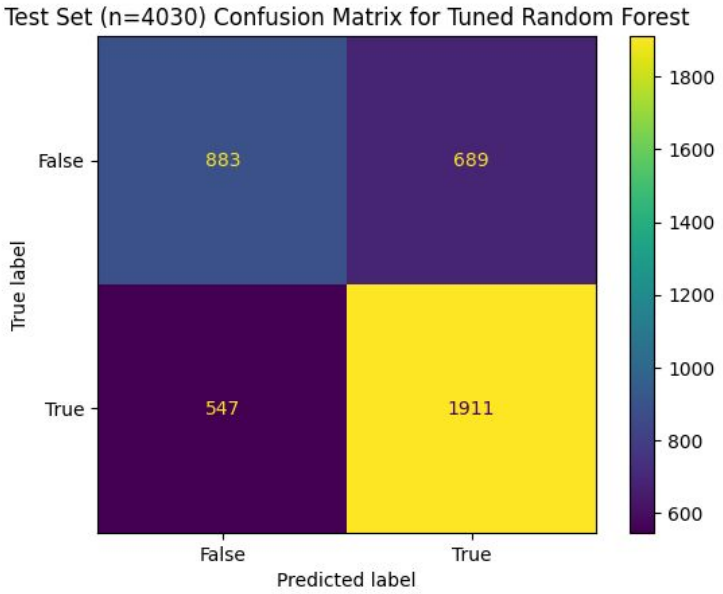


Tuned Threshold Cross-Validation (Weighted F1 Score)

Accuracy: 85%
Precision: 85%



Accuracy: 69%
Precision: 69%



Results and Interpretation

- Training Accuracy: 86% | Test Accuracy: 69%
- Model prioritizes Class-1 (Pass) accuracy
- Practical Use Cases:
  - Defensive play adjustment using real-time predictions
  - Enhanced NFL broadcast graphics for pre-snap analysis
  - Improve audience engagement during breaks

Conclusion

- RandomForest model predicts play types effectively
- Significant features: Pre-snap movement, position distances
- Opportunities for real-world applications (defense, broadcasts)
- Future Work: Expand features, enhance model performance

Questions?