Python Script Documentation: Logistic Regression on Tom Brady's Superbowl Win Prediction

1 Introduction

This document provides a comprehensive explanation of a Python script that performs logistic regression on NFL data to predict Superbowl wins. The script loads data, processes features, and trains logistic regression models for different time periods. It also evaluates the model performance and saves the results. This documentation covers the following aspects:

- Breakdown of the code.
- Explanation of loops, conditionals, and operations.
- Error handling mechanisms.
- A flowchart to visualize the code structure using TikZ.

2 Libraries Used

The script uses several Python libraries:

- `os`: Handles file system operations such as creating file paths and directories.
- `pandas`: Manages data loading, manipulation, and indexing.
- `sklearn.linear_model.LogisticRegression`: Implements the logistic regression algorithm for classification.

- `sklearn.preprocessing.StandardScaler`: Standardizes features by removing the mean and scaling to unit variance.
- `sklearn.metrics`: Provides functions to calculate evaluation metrics such as accuracy, precision, recall, F1 score, and confusion matrix.
- `matplotlib.pyplot`: Visualizes the confusion matrix and configures global plotting settings.

3 Main Functionality

The main purpose of the script is to:

- Load NFL data from a CSV file.
- Split the data into features and target (Superbowl wins).
- Train logistic regression models on different time periods (2002–2009, 2002–2019).
- Save the confusion matrix plots and evaluation metrics.

4 Code Breakdown

4.1 Logistic Regression Function

The `logistic_regression` function handles training and evaluation of the logistic regression model for a specified range of years.

- **Input Parameters:**
  - `train_year_start`: Start year for training data.
  - `train_year_end`: End year for training data.
  - `features`: DataFrame containing the feature variables.
  - `target`: Series containing the target variable (Superbowl Win).
  - `save_directory`: Path to save output files (plots and metrics).
- **Steps:**

- Split the data into training and testing sets based on the year range.
- Standardize the features using `StandardScaler`.
- Train a logistic regression model with `max_iter=1000`.
- Evaluate the model using the confusion matrix and save the result.
- Calculate evaluation metrics such as accuracy, precision, recall, and F1 score. Save the metrics to a text file.

4.2 Main Function

The `main` function orchestrates the data loading, preprocessing, and model training.

- Define paths for the data directory and result directory.
- Load data from a CSV file into a pandas DataFrame.
- Set the index to the `Season` column.
- Separate features from the target variable (Superbowl Win).
- Drop columns with NaN values from the features.
- Call the `logistic_regression` function to train models for two time periods (2002–2009 and 2002–2019).

5 Explanation of Loops and Nested Operations

Although there are no explicit loops in the script, there are nested operations and function calls:

- `StandardScaler`: Both `fit_transform` and `transform` methods are called to scale training and testing sets.
- `LogisticRegression`: The `fit` method is called to train the logistic regression model.
- The model predictions are evaluated using functions like `confusion_matrix`, `accuracy_score`, `precision_score`, etc.

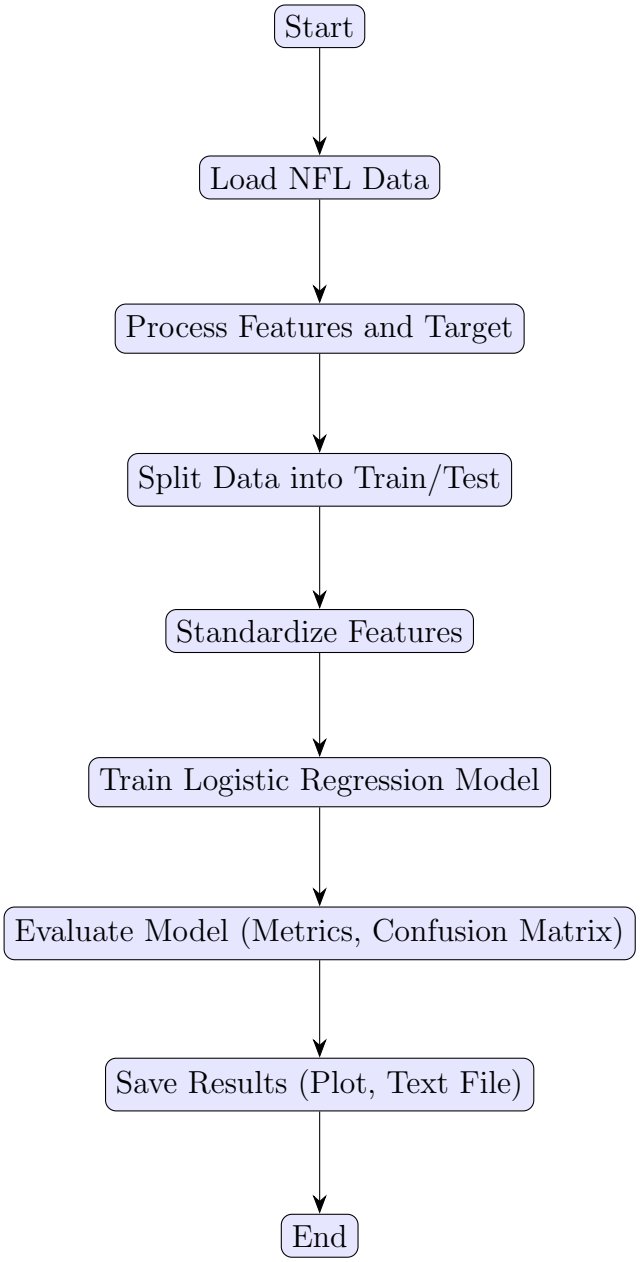
6 Error Handling

The script implicitly handles errors by:

- Using the `os` library to manage file paths, ensuring platform compatibility.
- Employing the `try-except` structure for file saving, but this is not explicitly coded in the provided script. For robust error handling, adding exceptions when reading files or saving results would be beneficial.

7 Flowchart

Below is a flowchart visualizing the structure of the script using the TikZ package:


Start
Load NFL Data
Process Features and Target
Split Data into Train/Test
Standardize Features
Train Logistic Regression Model
Evaluate Model (Metrics, Confusion Matrix)
Save Results (Plot, Text File)
End

8 Conclusion

The provided Python script efficiently trains logistic regression models on NFL data for predicting Superbowl wins. It processes features, evaluates models with various metrics, and saves the results for further analysis. This LaTeX documentation outlines the key functionalities, flow, and structure of the code.