Whether a NBA Team Will Enter The Playoff Since 2000

Background Information

The project aims to predict whether an NBA team will enter the playoffs based on its performance metrics during the regular season. The focus is on games played since the year 2000, as the modern era of basketball has seen a notable shift in scoring strategies, particularly with the increased popularity of the three-point shot. This change has significantly influenced offensive ratings and defensive ratings and makes data from the 21st century more representative of current trends.

The dataset used includes team name (team), season year (season), offensive rating (o_rtg), defensive rating (d_rtg), and net rating (n_rtg). The element net rating is used to evaluate a team's overall efficiency, both in terms of scoring and preventing the opposing team from scoring.

Problem Statement

The goal of the project is to classify whether an NBA team will enter the playoffs based on key statistical ratings from the regular season.

Hypothesis

1. Teams with higher net ratings (i.e., better offensive and defensive performance) are more likely to enter the playoffs.
2. Random Forest Classifier should be the most accurate and appropriate model to do prediction. Better accuracy value, Better Model.

Methods

The notebook uses a classification approach to predict playoff participation. Four different classification models were tested: Random Forest Classifier, Logistic Regression, and Support Vector Classifier (SVC), Decision Tree Classifier. Here's a summary of the methods used:

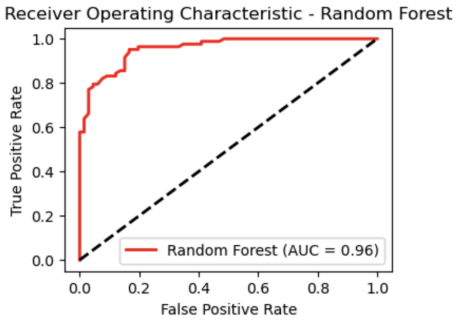
1. **Data Cleaning & Processing**:
    - The dataset is loaded and filtered to retain only the necessary columns: offensive rating, defensive rating, and net rating.
    - The project uses data from 2000 onwards to ensure the models are based on the modern era of NBA basketball.
2. **Feature Selection and Preprocessing**:
    - The **net rating** (calculated as offensive rating minus defensive rating) is used as the primary feature for model training.
    - **Season Year** and **Team Name** should be considered when training the model because they provide valuable contextual information that can help improve predictive accuracy.
        i. Season: Each season can have different dynamics due to factors like rule changes, player trades, and injuries. Including the SeasonYear feature allows the model to capture the effect of these year-specific differences.
        ii. Team: Including the TeamName as a feature allows the model to take into account the consistency and historical strength of specific teams, which is critical in predicting playoff qualification.
    - Data preprocessing involves standard scaling to normalize the feature set, ensuring that all models receive standardized input.

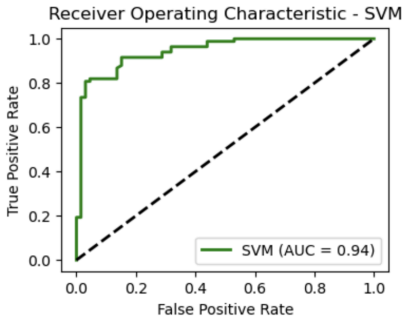
3. **Model Training and Evaluation**:
    - Four models were built using scikit-learn: Random Forest, Logistic Regression, SVC, and Decision Tree. Each model was trained and evaluated using metrics such as accuracy, classification reports, and confusion matrices.
    - A train-test split was used to divide the data into training and testing subsets. Cross-validation was performed to ensure the model results were not overly dependent on a particular data split.
    - The primary evaluation metrics included accuracy, precision, recall, and AUC (Area Under the Curve) for ROC and precision-recall curves.

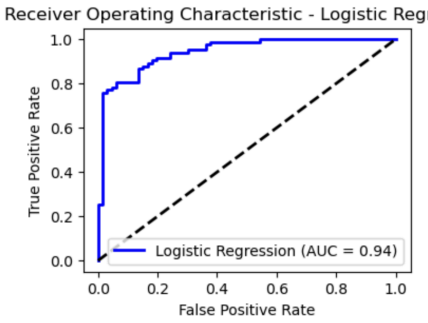
Results and Discussion

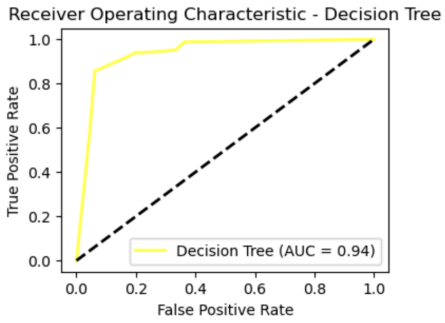
The Random Forest Classifier performed the best among the models tested, achieving the highest accuracy in predicting playoff participation, which is about 0.893. Other models approximately get 0.866, 0.866 and 0.879, which is a bit lower than the accuracy value Random Forest Classifier gets. We can know it from the reports of each model. I made this hypothesis because it is well-suited for capturing non-linear relationships and complex interactions between features such as Net Rating, Season Year, and Team Name. Random Forest is an ensemble learning method that constructs multiple decision trees during training and merges them to produce more accurate and stable predictions. It is also relatively robust to overfitting and generally provides good predictive performance in a variety of cases. For the given problem, where net rating, season year, and team name might have complex interactions, I think Random Forest is likely the best model as it can effectively capture non-linear patterns in the data.

We also can observe this difference in some diagrams, like ROC:









Random Forest AUC: 0.96

Others AUC: 0.94

Random Forest has the highest AUC (0.96), indicating the best overall performance among the models in distinguishing between playoff and non-playoff teams.

Decision Tree also has an AUC of 0.94 like Logistic Regression and SVM, but the curve is less consistent, suggesting potential overfitting or sensitivity to the dataset.

***Other evaluation methods can be found in the JupyterNotebook.**

However, we also observe that all accuracy values have not surpassed 90%. This probably means that an improvement in feature selection, model complexity, or additional external data might be necessary to achieve higher performance. Incorporating more relevant features, such as player

stats, injury reports, or advanced team metrics, could potentially boost accuracy and provide the model with more context for making predictions.

Overall, the net rating was found to be a reliable primary predictor of playoff qualification, which aligns with expectations as both offensive and defensive efficiencies are crucial for determining a team's success. The analysis also illustrated the value of modern data (post-2000) in understanding current basketball trends, especially with the growing importance of three-point shooting and other changes in team strategies.

Outside Resources

- NBA Stats Dataset (https://www.nba.com/stats/teams/advanced).
- Scikit-learn documentation: Used as a reference for model implementation and hyperparameter tuning (scikit-learn documentation).
- Matplotlib documentation (https://matplotlib.org/stable/tutorials/index.html)