Feature Engineering

Brian Fernando Aarush Maddela Nixon Tan
Sharona Yang

Wednesday 17th December, 2025

```
from nba_scraper.feature_engineering import *
```

1 Calculating Years of Experience for each player

```
# create a dictionary to store the first year a player appeared in the league
filepath = "data/nba_merged_stats/"
seasons = list(range(2002, 2026))
player_first_year = {}

# iterate through each season's data
for year in seasons:
    # load merged stats data for each season
    df = pd.read_csv(filepath + f"nba_merged_{year}.csv")
    # get the list of unique players in the current season
    players = df['Player'].dropna().unique()
    # store the first year a player appeared in the league
    for player in players:
        if player not in player_first_year:
            player_first_year[player] = year

# create 'Experience' feature for 2024 and 2025 seasons
df_2024 = pd.read_csv(filepath + f"nba_merged_2024.csv")
df_2024['Experience'] = df_2024['Player'].map(lambda x: (2024 + 1) - player_first_year.get(x


df_2025 = pd.read_csv(filepath + f"nba_merged_2025.csv")
df_2025['Experience'] = df_2025['Player'].map(lambda x: (2025 + 1) - player_first_year.get(x
```

2 One-Hot Encoding NBA Award Features

```
# create award -related features for the 2023 -2024 season
```

```
df_2024['Season'] = 2024
df_2024['NumOfAwards'] = df_2024['Awards'].apply(lambda x: len(x.split(',')) if pd.notna(x)
df_2024["All -Star"] = df_2024['Awards'].apply(lambda x: ("AS" in x) + 0 if pd.notna(x) else
df_2024['AwardWinner'] = df_2024['Awards'].apply(lambda x: check_award_winner(x) if pd.notna
df_2024['FirstTeam'] = df_2024['Awards'].apply(lambda x: all_nba_team_1(x) if pd.notna(x) el
df_2024['SecondTeam'] = df_2024['Awards'].apply(lambda x: all_nba_team_2(x) if pd.notna(x) e
df_2024['ThirdTeam'] = df_2024['Awards'].apply(lambda x: all_nba_team_3(x) if pd.notna(x) el
df_2024['DefTeam1'] = df_2024['Awards'].apply(lambda x: all_defensive_1(x) if pd.notna(x) el
df_2024['DefTeam2'] = df_2024['Awards'].apply(lambda x: all_defensive_2(x) if pd.notna(x) el

# repeat the same award -related feature creation for 2024 -2025 season
df_2025['Season'] = 2025
df_2025['NumOfAwards'] = df_2025['Awards'].apply(lambda x: len(x.split(',')) if pd.notna(x)
df_2025["All -Star"] = df_2025['Awards'].apply(lambda x: ("AS" in x) + 0 if pd.notna(x) else
df_2025['AwardWinner'] = df_2025['Awards'].apply(lambda x: check_award_winner(x) if pd.notna
df_2025['FirstTeam'] = df_2025['Awards'].apply(lambda x: all_nba_team_1(x) if pd.notna(x) el
df_2025['SecondTeam'] = df_2025['Awards'].apply(lambda x: all_nba_team_2(x) if pd.notna(x) e
df_2025['ThirdTeam'] = df_2025['Awards'].apply(lambda x: all_nba_team_3(x) if pd.notna(x) el
df_2025['DefTeam1'] = df_2025['Awards'].apply(lambda x: all_defensive_1(x) if pd.notna(x) el
df_2025['DefTeam2'] = df_2025['Awards'].apply(lambda x: all_defensive_2(x) if pd.notna(x) el
```

3 Adding in Current Year Salary and Next Year Salary features

```
# load player contracts for different seasons (2024, 2025, 2026)
filepath = "data/nba_player_contracts/"

salaries24 = player_contracts(filepath + 'nba_contracts_2024.csv')
salaries25 = player_contracts(filepath + 'nba_contracts_2025.csv')
salaries26 = player_contracts(filepath + 'nba_contracts_2026.csv')

# apply the team fixing function to the 2024 and 2025 dataframes
df_cleaned_2024 = fix_team_labels(df_2024)
df_cleaned_2025 = fix_team_labels(df_2025)

# merge player data with salary data for the 2024 season
full2024 = pd.merge(df_cleaned_2024, salaries24, on=['Player'])
# drop unnecessary columns from the merged dataframe
full2024 = full2024.drop(columns=['Rk', 'Tm', '2024 -25', '2025 -26', '2026 -27', '2027 -28'
# rename the salary column
full2024 = full2024.rename(columns={'2023 -24':"Salary"})
# update the salaries for 2024 -2025 season and merge with 2024 data
salaries25_updated = salaries25.loc[:, ['Player', '2024 -25', 'Guaranteed']]
salaries25_updated = salaries25_updated.rename(columns={'2024 -25':"Next_Year_Salary", "Guar
full2024_updated = pd.merge(full2024, salaries25_updated, on=['Player'], how='inner')
```

```
# save the updated data to a CSV file
full2024_updated.to_csv("data/final_2024_player.csv", index=False)

# repeat the same merging and cleaning process for the 2025 season
full2025 = pd.merge(df_cleaned_2025, salaries25, on=['Player'])
full2025 = full2025.drop(columns=['Rk', 'Tm', '2025 -26', '2026 -27', '2027 -28', '2028 -29'
full2025 = full2025.rename(columns={'2024 -25':"Salary"})
# update the salaries for 2024 -2025 season and merge with 2025 data
salaries26_updated = salaries26.loc[:, ['Player', '2025 -26', 'Guaranteed']]
salaries26_updated = salaries26_updated.rename(columns={'2025 -26':"Next_Year_Salary", "Guar
full2025_updated = pd.merge(full2025, salaries26_updated, on=['Player'], how='inner')

# save the updated data to a CSV file
full2025_updated.to_csv("data/final_2025_player.csv", index=False)
```