PREDICTING WINS WITH AI

FEATURES

01. Field Goal Attempts
02. Field Goal Percentage
03. 3 Point Attempts
04. 3 Point Percentage
05. Free Throw Attempts
06. Free Throw Percentage
07. Offensive Rebounds
08. Defensive Rebounds
09. Assists
10. Steals
11. Blocks
12. Turnovers
13. Personal Fouls

NBA MODEL

01. Retrieved data set from Kaggle

02. Found data spanning 2000-2023

03. Average of 30 teams per year resulting in 690 rows of data

WNBA MODEL

01. Created data set from the WNBA official statistics

02. Data spanning 2000-2023 *not including the 2013 season

03. Average of 13 teams per year resulting in 298 rows of data

STEP 1

CLEAN, FILTER, AND COMBINE DATASETS TO CREATE A DATAFRAME WITH DESIRED FEATURES

NBA DATA:

Isolated desired features in pergame_stats_total.csv, and used win percentages calculated from advanced_stats_total.csv

WNBA:

Isolated desired features in WNBA_statistics.csv

INCLUDED DATA

- ★ Games
- ★ Minutes Played
- ★ Field Goals
- ★ Field Goal Attempts
- ★ Field Goal Percent
- ★ 3 Pointers
- ★ 3 Point Attempts
- ★ 3 Point Percentage
- ★ 2 Pointers
- ★ 2 Point Attempts
- ★ 2 Point Percentage
- ★ Free Throws
- ★ Free Throw Attempts
- ★ Free Throw Percentage
- ★ Offensive Rebounds
- ★ Defensive Rebounds
- ★ Total Rebounds
- ★ Assists
- ★ Steals
- ★ Blocks
- ★ Turnovers
- ★ Personal Fouls
- ★ Points
- ★ Year

NBA DATA FRAME CODE

```python
def NBA_df_config():
    NBA_df = pd.read_csv("pergame_stats_total.csv")

    NBA_df = NBA_df[["Team", "FGA", "FG_Percent", "3PA", "3P_Percent", "FTA", "FT_Percent", "ORB", "DRB", "AST",
                    "STL", "BLK", "TOV", "PF", "Year"]]

    NBA_df = NBA_df[NBA_df["Team"] != "League Average"]

    win_percentage_df = pd.read_csv("advanced_stats_total.csv")
    win_percentage_df = win_percentage_df[["Team", "W", "L", "Year"]]

    winPctTeam = {}
    # don't want first row bc just column headers
    for index, row in win_percentage_df.iterrows():

        if (row.iloc[0] != "League Average") and (row.iloc[0] != "Team"):

            team = row.iloc[0]
            wins = int(row.iloc[1])
            losses = int(row.iloc[2])
            year = int(row.iloc[3])
            win_pct = wins / (wins + losses)
            teamYr = (team, year)
            winPctTeam[teamYr] = win_pct


    NBA_df["WIN%"] = 0.0

    # Update NBA_df to include win percentages
    for index, row in NBA_df.iterrows():
        year = NBA_df.at[index, "Year"]
        if (row.iloc[0], year) in winPctTeam:
            NBA_df.loc[index, "WIN%"] = winPctTeam[(row["Team"], year)]

    # Check for missing win percentages
    NBA_df = NBA_df.dropna(subset=["WIN%"])

    # remove year column so df is identical to wNBA
    NBA_df = NBA_df.drop(columns=['Year'])

    return NBA_df
```

WNBA DATA FRAME CODE

```python
def wNBA_df_config():
    wNBA_df = pd.read_csv("WNBA Statistics.csv")

    wNBA_df = wNBA_df[["Team", "FGA", "FG_Percent", "3PA", "3P_Percent", "FTA", "FT_Percent", "ORB", "DRB", "AST",
                        "STL", "BLK", "TOV", "PF", "WIN%"]]

    wNBA_df = wNBA_df[wNBA_df["Team"] != "League Average"]

    return wNBA_df
```

STEP 2

RUN A REGRESSION ON FEATURES OF BOTH DATA SETS TO TRACK R^2 VALUES AND COEFFICIENTS

TO CREATE OUR REGRESSION FUNCTION:

1. Scaled each data set and split them into training (80%) and test (20%) data
2. Ran regression on each feature for each data set
3. Found coefficients and R2 values for each feature
4. Returned data frame containing these values

FEATURE REGRESSION CODE

```python
def regression(df):

    # Run Regression
    X = df[["FGA", "FG_Percent", "3PA", "3P_Percent", "FTA", "FT_Percent", "ORB", "DRB", "AST",
            "STL", "BLK", "TOV", "PF"]]
    y = df['WIN%']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # standardize features of X_train and X_test

    scaler = StandardScaler()

    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)

    # track r2 for each feature
    r2Final = []

    # track coefficients
    coefficients = []

    for i, feature in enumerate(X.columns):

        model = LinearRegression()

        # run regression on single feature at a time, have to reshape vectors
        x_train_feature = np.array(X_train[:, i]).reshape(-1, 1)
        x_test_feature = np.array(X_test[:, i]).reshape(-1, 1)

        model.fit(x_train_feature, y_train)

        # compare prediction with actual values using r2_score method
        prediction = model.predict(x_test_feature)
        r2 = r2_score(y_test, prediction)

        coeff = model.coef_

        r2Final.append(r2)
        coefficients.append(coeff[0])

    overallModel = LinearRegression()
    overallModel.fit(X_train, y_train)
```

```python
    overall_predictions = overallModel.predict(X_test)

    overall_r2 = r2_score(y_test, overall_predictions)
    overall_mse = mean_squared_error(y_test, overall_predictions)


    # make overall dataframe
    overallData = {'Metric': ['R2 Score', 'Mean Squared Error'], '': [overall_r2, overall_mse]}
    overallDataFrame = pd.DataFrame(overallData)


    # make featuredataframe
    featureData = {'Feature': X.columns, 'R2': r2Final, 'Coef': coefficients}

    featureDataFrame = pd.DataFrame(featureData)
    return (featureDataFrame, overallDataFrame)


def main():
    print("\n")
    NBA_df = NBA_df_config()
    wNBA_df = wNBA_df_config()

    NBA_features = regression(NBA_df)[0]
    WNBA_features = regression(wNBA_df)[0]

    r2_Feature_Comparison = NBA_features.merge(WNBA_features, on="Feature", suffixes=("_NBA", "_WNBA"))
    print(r2_Feature_Comparison)
    print("\n")


    NBA_overall = regression(NBA_df)[1]
    WNBA_overall = regression(wNBA_df)[1]

    r2_Overall_Comparison = NBA_overall.merge(WNBA_overall, on="Metric", suffixes=("NBA", "WNBA"))
    print(r2_Overall_Comparison)


main()
```

OUTPUT

	Feature	R2_NBA	Coef_NBA	R2_WNBA	Coef_WNBA
0	FGA	-0.035074	-0.007146	-0.000057	0.011900
1	FG_Percent	0.196626	0.086447	0.112346	0.105436
2	3PA	-0.033038	0.016223	-0.003278	0.002038
3	3P_Percent	0.091817	0.076641	0.264033	0.064155
4	FTA	-0.021621	0.009055	0.034658	0.025791
5	FT_Percent	-0.012429	0.025109	0.039795	0.037329
6	ORB	-0.022305	-0.021821	-0.003495	-0.002729
7	DRB	0.032339	0.049224	0.119867	0.055886
8	AST	0.060491	0.044339	-0.010855	0.073733
9	STL	-0.018423	0.020569	0.044954	0.034679
10	BLK	0.010849	0.038213	-0.109414	0.050492
11	TOV	0.084080	-0.053861	-0.116616	-0.058413
12	PF	-0.086126	-0.036560	0.076505	-0.045965

IMPORTANT COEFFICIENTS

NBA		WNBA	
.087	Field Goal %	.105	Field Goal %
.077	3 Point %	.073	Assist
-.054	Turnovers	.064	3 Point %
.049	Defensive Rebounds	-.058	Turnovers

IMPORTANT R^2 VALUES

NBA		WNBA	
.197	Field Goal %	.264	3 Point %
.092	3 Point %	.120	Defensive Rebounds
.084	Turnovers	.112	Field Goal %
.060	Assists	.076	Personal Fouls

COEFFICIENT & R^2 ANALYSIS

MOST IMPACTFUL NBA FEATURE IN PREDICTING WIN %

1. Field Goal Percentage
2. 3 Point Percentage

MOST IMPACTFUL WNBA FEATURE IN PREDICTING WIN %

1. Field Goal Percentage
2. Assists

FEATURES EXPLAINING WIN % VARIATION

1. Field Goal Percentage
2. 3 Point Percentage

FEATURES EXPLAINING WIN % VARIATION

1. 3 Point Percentage
2. Defensive Rebounds

STEP 3

RUN OVERALL LOGISTIC REGRESSION MODEL ON NBA + WNBA

TO CREATE OUR OVERALL REGRESSION FUNCTION:

1. Scaled data and split into training (80%) and test (20%) data
2. Ran regression
3. Used predictions to calculate R2 score and MSE for NBA and WNBA model
4. Returned data frame containing these values

OVERALL REGRESSION CODE

```python
def regression(df):

    # Run Regression
    X = df[["FGA", "FG_Percent", "3PA", "3P_Percent", "FTA", "FT_Percent", "ORB", "DRB", "AST",
            "STL", "BLK", "TOV", "PF"]]
    y = df['WIN%']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # standardize features of X_train and X_test

    scaler = StandardScaler()

    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)

    # track r2 for each feature
    r2Final = []

    # track coefficients
    coefficients = []

    for i, feature in enumerate(X.columns):

        model = LinearRegression()

        # run regression on single feature at a time, have to reshape vectors
        x_train_feature = np.array(X_train[:, i]).reshape(-1, 1)
        x_test_feature = np.array(X_test[:, i]).reshape(-1, 1)

        model.fit(x_train_feature, y_train)

        # compare prediction with actual values using r2_score method
        prediction = model.predict(x_test_feature)
        r2 = r2_score(y_test, prediction)

        coeff = model.coef_

        r2Final.append(r2)
        coefficients.append(coeff[0])

    overallModel = LinearRegression()
    overallModel.fit(X_train, y_train)
```

```python
    overall_predictions = overallModel.predict(X_test)

    overall_r2 = r2_score(y_test, overall_predictions)
    overall_mse = mean_squared_error(y_test, overall_predictions)


    # make overall dataframe
    overallData = {'Metric': ['R2 Score', 'Mean Squared Error'], '': [overall_r2, overall_mse]}
    overallDataFrame = pd.DataFrame(overallData)


    # make featuredataframe
    featureData = {'Feature': X.columns, 'R2': r2Final, 'Coef': coefficients}

    featureDataFrame = pd.DataFrame(featureData)
    return (featureDataFrame, overallDataFrame)


def main():
    print("\n")
    NBA_df = NBA_df_config()
    wNBA_df = wNBA_df_config()

    NBA_features = regression(NBA_df)[0]
    WNBA_features = regression(wNBA_df)[0]

    r2_Feature_Comparison = NBA_features.merge(WNBA_features, on="Feature", suffixes=("_NBA", "_WNBA"))
    print(r2_Feature_Comparison)
    print("\n")


    NBA_overall = regression(NBA_df)[1]
    WNBA_overall = regression(wNBA_df)[1]

    r2_Overall_Comparison = NBA_overall.merge(WNBA_overall, on="Metric", suffixes=("NBA", "WNBA"))
    print(r2_Overall_Comparison)


main()
```

OUTPUT

	Metric	NBA	WNBA
0	R2 Score	0.770295	0.637932
1	MSE	0.004629	0.007805

R^2 & MSE ANALYSIS

NBA

R^2: 0.770

MSE: 0.005

WNBA

R^2: 0.638

MSE: 0.008

1. NBA has higher R^2: model explains more variability in win %
2. NBA has lower MSE: model predictions are more accurate

WHY? → More training data for NBA vs WNBA (more teams)

STEP 4

TEST MODEL ON 2024 WNBA DATA

PREDICTED WIN %:

Liberty: 87
Lynx: 82
Sun: 68
Aces: 90
Storm: 43
Fever: 50
Mercury: 48
Dream: 43
Mystics: 29
Sky: 23
Wings: 8
Sparks: 30

CORRECT WIN %:

Liberty: 80
Lynx: 75
Sun: 70
Aces: 68
Storm: 63
Fever: 50
Mercury: 48
Dream: 36
Mystics: 35
Sky: 33
Wings: 23
Sparks: 20

% ERROR:

Liberty: 8.6
Lynx: 8.7
Sun: 3.2
Aces: 33.6
Storm: 30.6
Fever: 0.01
Mercury: 1.6
Dream: 14.5
Mystics: 17.5
Sky: 29.7
Wings: 66.3
Sparks: 50.9

CONCLUSIONS

THREE MAJOR TAKEAWAYS

1. FIELD GOAL PERCENTAGE SEEMED TO HAVE THE STRONGEST CORRELATION WITH WIN PERCENTAGE IN BOTH THE NBA AND WNBA

2. WIN % VARIATION IS CAUSED MORE SO BY 3 POINT PERCENTAGE IN THE WNBA, AND FIELD GOAL PERCENTAGE IN THE NBA

3. OUR NBA MODEL IS MORE ACCURATE THAN OUR WNBA MODEL IN PREDICTING WIN PERCENTAGE

LIMITATIONS

1. Amount of Data
    a. Used 23 years of data for each league
    b. More data → better model
2. Features Used
    a. Model only considers some features
    b. Not injuries, coaching, etc
3. Correlation vs Causation
    a. Regression shows correlation, not causation

Next Steps

1. Use more years of NBA + WNBA data
2. Include more features in each model
3. Expand to other sports?

SOURCES

1. https://stats.wnba.com/teams/traditional/?sort=W_PCT&dir=-1
2. https://www.nba.com/stats/teams/traditional?dir=A&sort=W_PCT