

# Prediction of NFL Scores and Spread Optimization

Ashley Lauren. Mersman

Northwest Missouri State University, Maryville MO 64468, USA  
S2571660@nwmissouri.edu

**Abstract.** The abstract should briefly summarize the contents of the paper in 150–250 words.

**Keywords:** sports · betting · machine learning · NFL

## 1 Introduction

In 2023, approximately 28% of the American public was expected to bet on the NFL. That's 73 million people for an increase of almost 60% from the previous year. Sports betting is legal in 34 states and counting and is a multi-billion dollar industry. [2]

Being able to predict game outcomes and optimize the spread is highly profitable for the industry. The spreads must be close enough to incentivize bettors but also large enough to maximize profits and cover any losses.

Game information (date, season, week, teams, and scores), betting information (favorite to win, spread, over/under), stadium information (stadium, location, elevation, field type, and stadium type) and weather information (temperature, wind speed, and humidity).

### 1.1 Goals of this Research

The goal of this project is to create a new predictive model for game outcomes and score differentials.

### 1.2 Process

Data

#### 1. Data Collection

- (a) A curated dataset from Kaggle.com was used that relied on sources for weather, NFL team information, betting information, and game information from the past 50+ years. [4]

#### 2. Data Cleaning

- (a) The data will be cleaned to filter out unnecessary columns leaving the following, game information (date, season, week, teams, and scores), betting information (favorite to win, spread, over/under), stadium information (stadium, location, elevation, field type, and stadium type) and weather information (temperature, wind speed, and humidity).
  - (b) Columns will have their data types checked and corrected.
  - (c) Check for and correct any duplicates.
3. Data Exploration
- (a) Check for correlations and distributions.

## Machine Learning

- 1. Pre-processing
- 2. Model Selection and Parameter Tuning
- 3. Model Analysis

## Conclusions

# 2 Data

## 2.1 Data Collection

A curated dataset on Kaggle.com by Spreadspoke, a sports data analysis company, was used as the basis of this project. Spreadspoke used a variety of sources, including but not limited to: ESPN, NFL.com, NOAA, NFLweather.com, Pro-Football.com and multiple others referenced on the original dataset page. The dataset is refreshed on a weekly basis. The data is available as multiple CSV's and the R file used to curate the dataset. [4]

The dataset time frame is 1966 for NFL game results and 1979 for betting odds data.

The kaggle package was used with the kaggle api to download the dataset. [1]

nfl\_stadiums.csv contains data about the 120 stadiums each game has been played. This data goes beyond simply NFL stadiums due to internationally played games and other special games. Variables:

- 1. stadium\_name
- 2. stadium\_location: city, state of the stadium
- 3. stadium\_open: the year the stadium opened
- 4. stadium\_close: the year the stadium closed
- 5. stadium\_type: weather type (indoor, outdoor, retractable)
- 6. stadium\_address
- 7. stadium\_weather\_station\_zip-code: the zip-code used for weather data collection
- 8. stadium\_weather\_type: weather category based on average temperature
- 9. stadium\_capacity: stadium seating maximum

10. stadium\_surface: field type (Grass, Turf, Field-Turf, Hellas Matrix Turf, Grass, Turf (1969-1970), Grass, Turf (1970-1971), Grass, Turf (1971-1974))
11. stadium\_weather\_station: weather station ID for NOAA data
12. stadium\_weather\_station\_name
13. stadium\_latitude
14. stadium\_longitude
15. stadium\_azimuthangle: angle of the stadium from North
16. stadium\_elevation

nfl\_teams.csv is a datafile containing information about the specific NFL teams. There are 44 entries due to teams moving cities, changing names, and other modifications. Variables:

1. team\_name
2. team\_name\_short
3. team\_id
4. team\_id\_pfr: team id on Profootball-reference.com
5. team\_conference
6. team\_division
7. team\_conference\_pre2022
8. team\_division\_pre2022

The game data in spreadspoke\_scores.csv contains game data since the 1966 season totaling 13,788 games. It has game information, weather information, and betting information Variables:

1. schedule\_date: date game played
2. schedule\_season: game season
3. schedule\_week: week of season
4. schedule\_playoff: boolean value, is this a playoff game
5. team\_home: home team
6. score\_home: home team score
7. score\_away: away team score
8. team\_away: away team
9. team\_favorite\_id: favorite team to win
10. spread\_favorite: spread
11. over\_under\_line: over under
12. stadium: stadium name
13. stadium\_neutral: boolean, is the stadium a neutral site
14. weather\_temperature
15. weather\_wind\_mph
16. weather\_humidity
17. weather\_detail: precipitation detail

The season time information(schedule\_date, schedule\_season, schedule\_week, and schedule\_playoff), scoring and team information (team\_home, team\_away, score\_home, score\_away), and conditions (stadium, stadium\_neutral, stadium\_type, stadium\_surface, stadium\_elevation, weather\_temperature,

weather\_wind\_mph, weather\_humidity, weather\_detail) will be used to predict the game outcome. The betting information (team\_favorite\_id, spread\_favorite, and over\_under\_line) will be used to optimize the spread. Winner will be added by comparing the team\_home and score\_home with the team\_away and score\_away columns.

## 2.2 Data Cleaning

The data cleaning process started by inspecting the datasets using Pandas `pd.info()` The commas were removed from the stadium\_capacity variable using `pd.replace()` and the datatype for stadium\_capacity, stadium\_open, and stadium\_close were changed to Int64 using `pd.astype()`.

```
stadiums.stadium_capacity = stadiums.stadium_capacity.replace(',', '', regex=True)
stadiums.stadium_capacity = stadiums.stadium_capacity.astype("Int64")
stadiums.stadium_open = stadiums.stadium_open.astype("Int64")
stadiums.stadium_close = stadiums.stadium_close.astype("Int64")
```

A column for "winner" was created in the dataset for scores by iterating through the dataframe and comparing the score\_home with the score\_away. The team with the larger score was selected as winner and appended to a list of winners. In the case the scores are equal Tie is appended. The list is then converted to a column in dataframe.

```
#establish who won each game
winners = []
```

```
#iterate through games compariong scores to determine winners, append the winner to the lis
for i,v in games.score_home.items():
    if games.score_home[i] > games.score_away[i]:
        winners.append(games.team_home[i])

    elif games.score_away[i]>games.score_home[i]:
        winners.append(games.team_away[i])

    else:
        winners.append("Tie")
```

```
#convert list to column
games["winner"] = winners
```

The spread is a point differential that quantifies the margin by which a team is expected to win or lose. A spread of +7 for a home team means that the home team is expected to lose by 7 points. The favored team must win by more than 7 points in order for the bettor to win a bet for the favored team. [3] The spread\_favorite was set up conversely to what is expected by betting lines and had missing values. Therefore a new spread (score\_diff) was created to act as the spread. A score\_total variable was also created.

```
games['score_difference'] = games['score_away'] - games['score_home']
games['score_total'] = games['score_away'] + games['score_home']
```

The datasets for the score information and stadium information were then merged into one dataframe using `pd.merge()`.

```
merged_df = games.merge(stadiums, left_on="stadium",
                        right_on="stadium_name", suffixes=['_x', '_y'], how="left")
```

The dataframe columns were selected and reordered. Null values were then examined. All rows with missing scores for either `score_home` or `score_away` were dropped. The null values under `weather_detail` were replaced with "No Precip" since the column is used to indicate additional weather information, it is assumed then that the lack of data means lack of precipitation. Lastly, due to the large number of missing values for `stadium_surface` this column was dropped. The mean value for `weather_humidity`, `weather_wind_mph`, and `stadium_elevation` were used in place of the null values.

```
merged_df = merged_df[['schedule_date', 'schedule_season', 'schedule_week', 'schedule_playo
                        'team_home', 'team_away', 'score_home', 'score_away', 'winner', 'sco
                        , 'stadium', 'stadium_neutral', 'stadium_type', 'stadium_elevation',
                        'weather_temperature', 'weather_wind_mph', 'weather_detail', 'weather
```

```
merged_df = merged_df.dropna(subset=['score_home'])
merged_df = merged_df.dropna(subset=['stadium_type'])
```

```
mean_humidity = merged_df['weather_humidity'].mean()
merged_df["weather_humidity"].fillna(mean_humidity, inplace = True)
```

```
mean_temp = merged_df['weather_temperature'].mean()
merged_df["weather_temperature"].fillna(mean_temp, inplace = True)
```

```
mean_wind = merged_df['weather_wind_mph'].mean()
merged_df["weather_wind_mph"].fillna(mean_wind, inplace = True)
```

```
mean_elevation = merged_df['stadium_elevation'].mean()
merged_df["stadium_elevation"].fillna(mean_elevation, inplace = True)
```

```
merged_df['weather_detail'].fillna('No Precip', inplace = True)
```

The data was checked for duplicates using `pd.duplicated().sum()`. No duplicated values were detected.

The cleaned dataset has 13,558 rows and 20 columns.

**Table 1.** Data Variables and Definitions

Variable	Description	Data Type
schedule_date	Date of the game	Object
schedule_season	Year of Football Season	Integer
schedule_week	Week of season	Object
schedule_playoff	True for Playoff, False for Regular Season	Boolean
team_home	Home Team	Object
team_away	Away Team	Object
score_home	Home Team Score	Float
score_away	Away Team Score	Float
winner	Winning Team	Object
score_difference	Difference Between Away Team Score and Home Team Score	Float
score_total	Total Points Scored	Float
stadium	Stadium Name	Object
stadium_neutral	False for Bias, True for Neutral Site	Boolean
stadium_type	Outdoor, Indoor, Or Covered Stadium	Object
stadium_elevation	Elevation of Stadium	Float
weather_temperature	Temperature at Game, F	Float
weather_wind_mph	Wind Speed at Game, mph	Float
weather_detail	Precipitation at Game	Object
weather_humidity	Humidity at Game, %	Float

Finally, the distribution of the columns was reviewed as a final check for outliers, null values, or strange data that would require further examination. `pd.describe(include='all')`

The dependent variable will be the score difference

The raw files for this project can be found at The Python notebook and other files used for this project can be found at [https://github.com/AMersman/capstone\\_NFL](https://github.com/AMersman/capstone_NFL).

## 2.3 Data Exploration

**Statistics and Verification** Data exploration started by confirming the data was cleaned and missing values were imputed correctly. The first rows of the dataframe were inspected. `df.head()` The datatypes were verified once more. `df.info()` and the summary statistics were viewed using `df.summary(include='all')`.

The Patriots are the winning-est team. Outdoor stadiums are most common. Most games do not have precipitation.

The mean Score for hometeam is 22.76 and for the awayteam is 20.09; there is a home team advantage but it's not large (2.67 points and the mean spread\_favorite is 2.66) The mean number of points scored overall is 42.85. There is also not much difference between the mean and the median of these values.

The distribution of the variables was visualized to check for spread and outliers using `df.hist(figsize=(10,10))`

	schedule_date	schedule_season	schedule_week	schedule_playoff	team_home	team_away	score_home	score_away	winner	spread_favorite
count	11098	11098.000000	11098.000000	11098.000000	11098	11098	11098.000000	11098.000000	11098	11098.000000
unique	2117	NaN	NaN	NaN	43	43	NaN	NaN	44	NaN
top	1/3/2010	NaN	NaN	NaN	San Francisco 49ers	Pittsburgh Steelers	NaN	NaN	New England Patriots	NaN
freq	16	NaN	NaN	NaN	363	374	NaN	NaN	451	NaN
mean	NaN	2001.582389	9.505216	0.043782	NaN	NaN	23.782788	20.282711	NaN	-2.656193
std	NaN	12.891542	5.268300	0.204641	NaN	NaN	10.304789	10.070002	NaN	5.803790
min	NaN	1996.000000	1.000000	0.000000	NaN	NaN	0.000000	0.000000	NaN	-26.000000
25%	NaN	1999.000000	5.000000	0.000000	NaN	NaN	10.000000	10.000000	NaN	-6.000000
50%	NaN	2002.000000	10.000000	0.000000	NaN	NaN	20.000000	20.000000	NaN	-3.000000
75%	NaN	2013.000000	14.000000	0.000000	NaN	NaN	30.000000	27.000000	NaN	2.000000
max	NaN	2023.000000	22.000000	1.000000	NaN	NaN	70.000000	69.000000	NaN	16.000000

Fig. 1. Summary Statistics for Cleaned Dataset

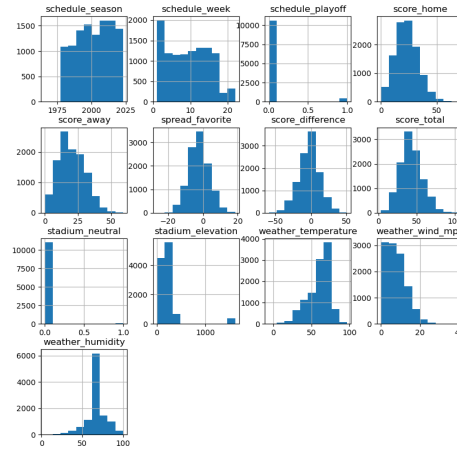


Fig. 2. Distribution of Quantitative Variables

While not completely symmetrical there is little skew to most variables. It should be noted that two of the variables are actually Boolean representations and not continuous variables, hence the distributions. `schedule_season` and `schedule_week` are relatively uniform which makes sense for the variable. The increase in number of games played per season makes up the jump in `schedule_season` distribution.

**Visualizations** The differences in `score_difference` across multiple variables was visualized by iterating through the columns to create bar charts of the variables.

```
'''Check median score difference across selected variables'''
def plot_median_diff(column_name):
    items = df[column_name].unique()
    med_total_list = []

    for item in items:
        med_total = df[df[column]==item]['score_difference'].median()
        med_total_list.append(med_total)
```

```

plt.figure(figsize=(10, 6))
bars = plt.bar(items, med_total_list, align='center')
plt.xlabel(column_name)
plt.ylabel('Median Score Difference')
plt.title(f'Median Score Difference vs {column_name}')
plt.xticks(rotation=90)
plt.tight_layout()

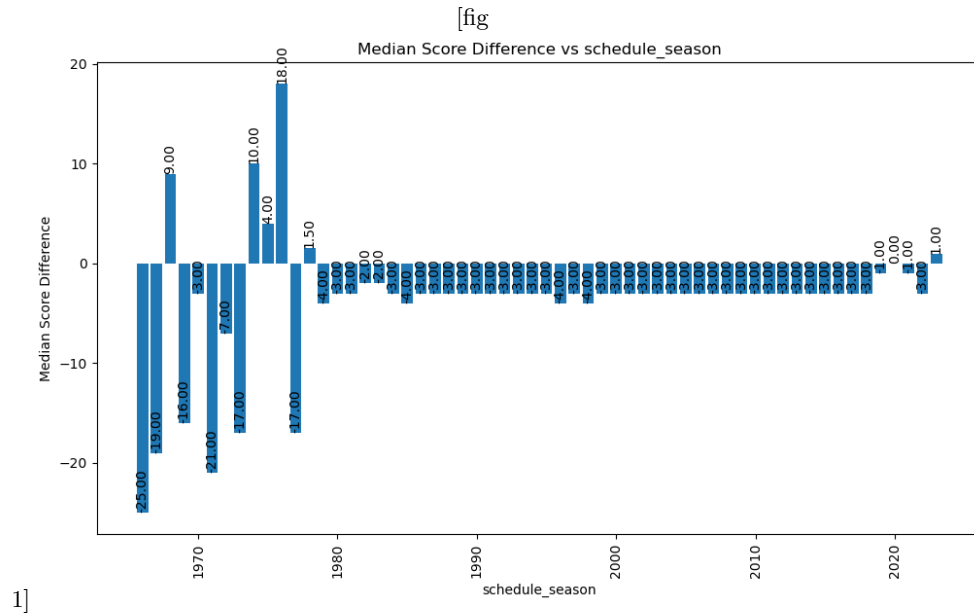
for bar, label in zip(bars, med_total_list):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), f'{label:.2f}', ha='center')

plt.show()

columns_of_interest = ['schedule_season', 'team_home', 'team_away', 'weather_temperature',

for column in columns_of_interest:
    plot_median_diff(column)

```



**Fig. 3.** Score Difference by Season

The score differences have stabilized and are very small (-3 favoring the home team since the 1980's. This is by design with rule changes favoring close games



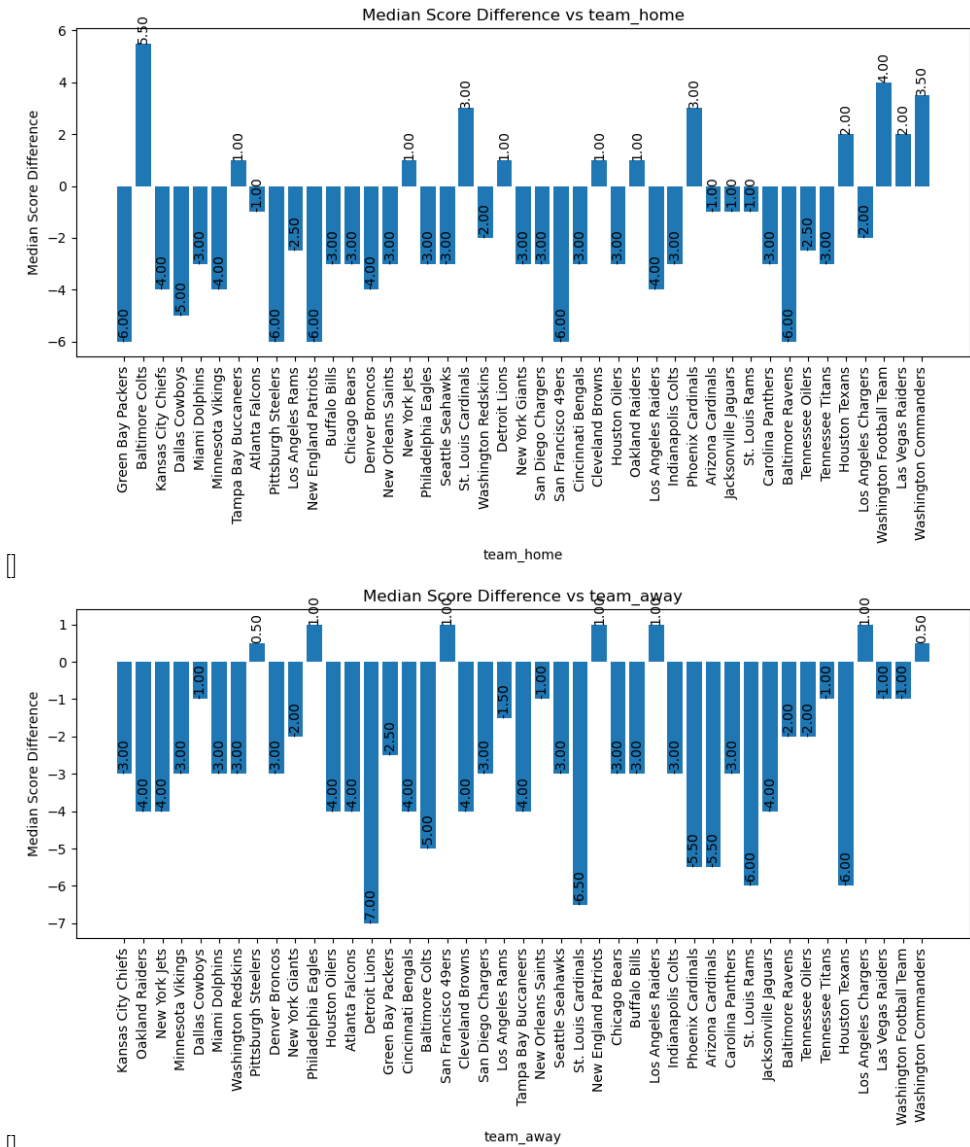
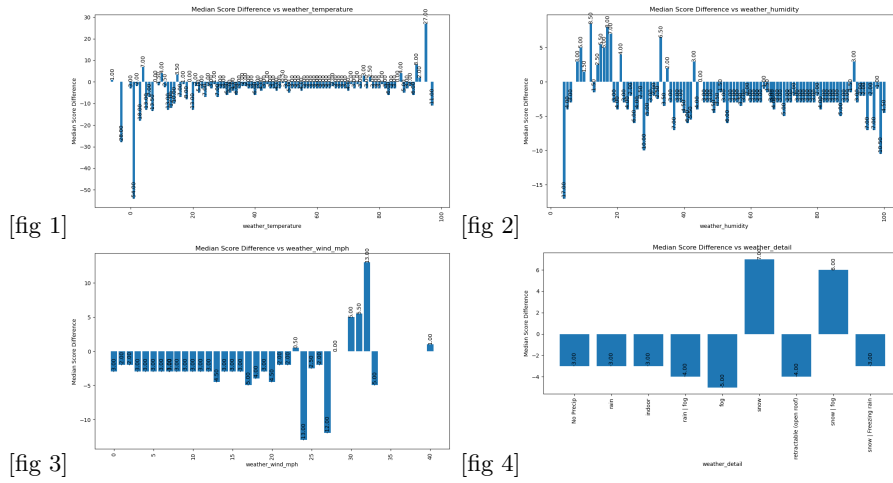


Fig. 4. Score Difference by Home and Away Team



**Fig. 5.** Score Difference by Weather Conditions

to increase interest and viewership. Weather extremes have an effect on score difference and oddly snow/fog seem to favor the away team which is unexpected.

The same process was used to visualize the score\_total across the same variables.

```
'''Check median score total across selected variables'''
def plot_median_total(column_name):
    items = df[column_name].unique()
    med_total_list = []

    for item in items:
        med_total = df[df[column]==item]['score_total'].median()
        med_total_list.append(med_total)

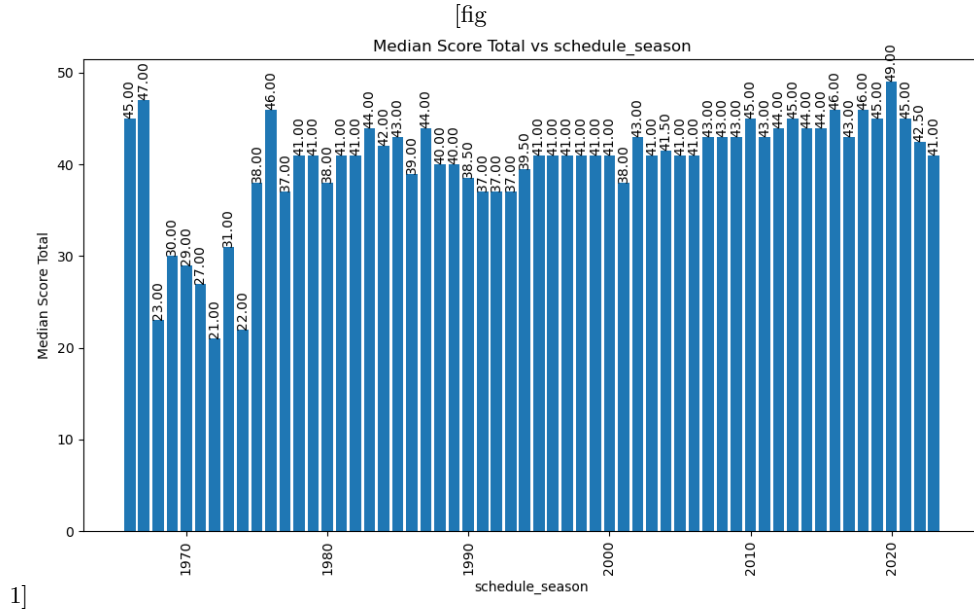
    plt.figure(figsize=(10, 6))
    bars = plt.bar(items, med_total_list, align='center')
    plt.xlabel(column_name)
    plt.ylabel('Median Score Total')
    plt.title(f'Median Score Total vs {column_name}')
    plt.xticks(rotation=90)
    plt.tight_layout()

    for bar, label in zip(bars, med_total_list):
        plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), f'{label:.2f}',
                 ha='center',
                 va='bottom', rotation = 90)

    plt.show()

columns_of_interest = ['schedule_season', 'team_home', 'team_away', 'weather_humidity',
                       'weather_temperature', 'weather_detail', 'weather_wind_mph' ]

for column in columns_of_interest:
    plot_median_total(column)
```

**Fig. 6.** Score Total by Season

Some interesting patterns emerged while viewing the data. Contrary to first assumptions, lower temperatures/lower humidity typically have higher score totals as do snow/freezing rain games. As previously mentioned these games favor the away team but they are also higher scoring overall.

As expected though, score totals go down as wind speed increases. This makes sense when you consider the ways in which points are scored in football. The higher wind speeds make scoring points more difficult because it reduces the accuracy of the field-goal kicker to score. It also reduces the accuracy of the passing game and punting.

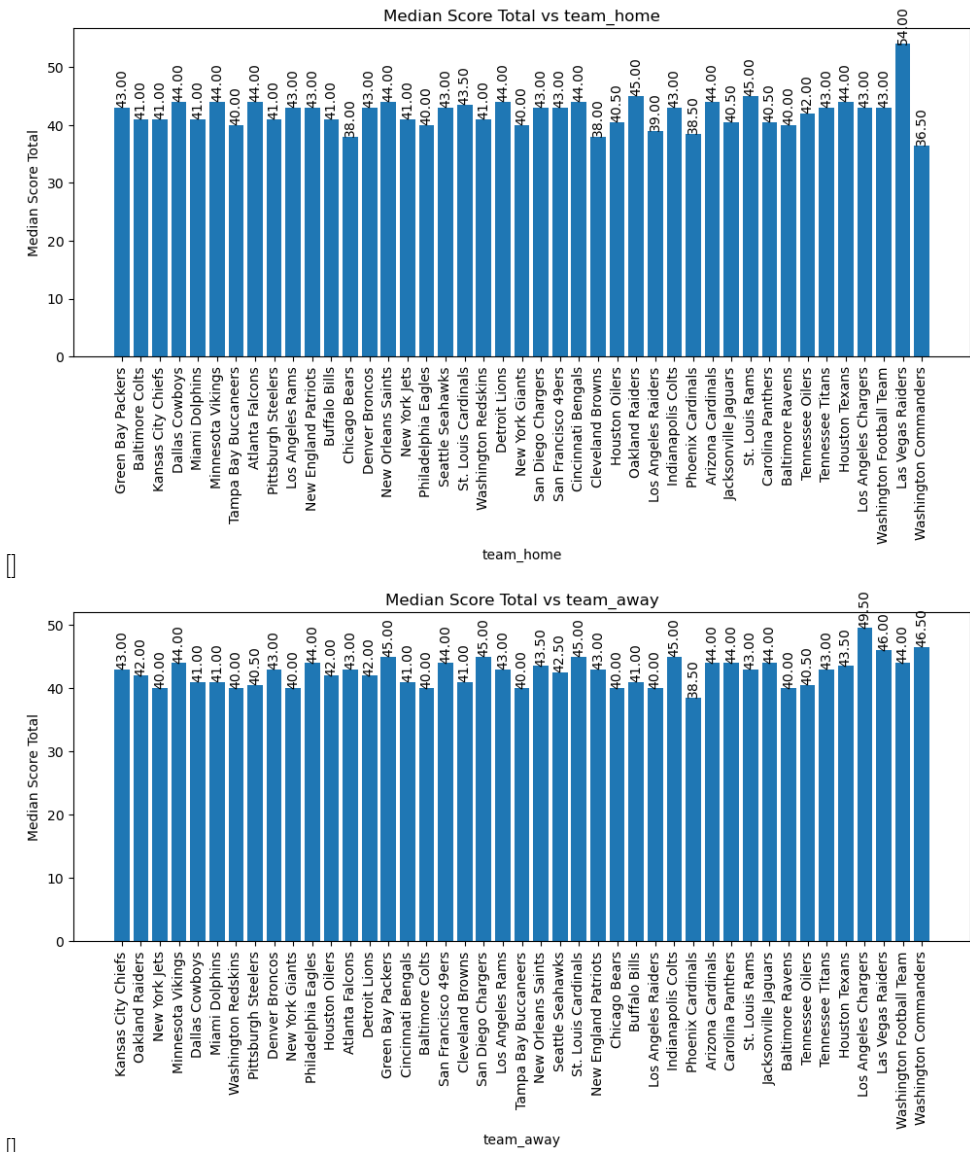


Fig. 7. Score Total by Home and Away Team



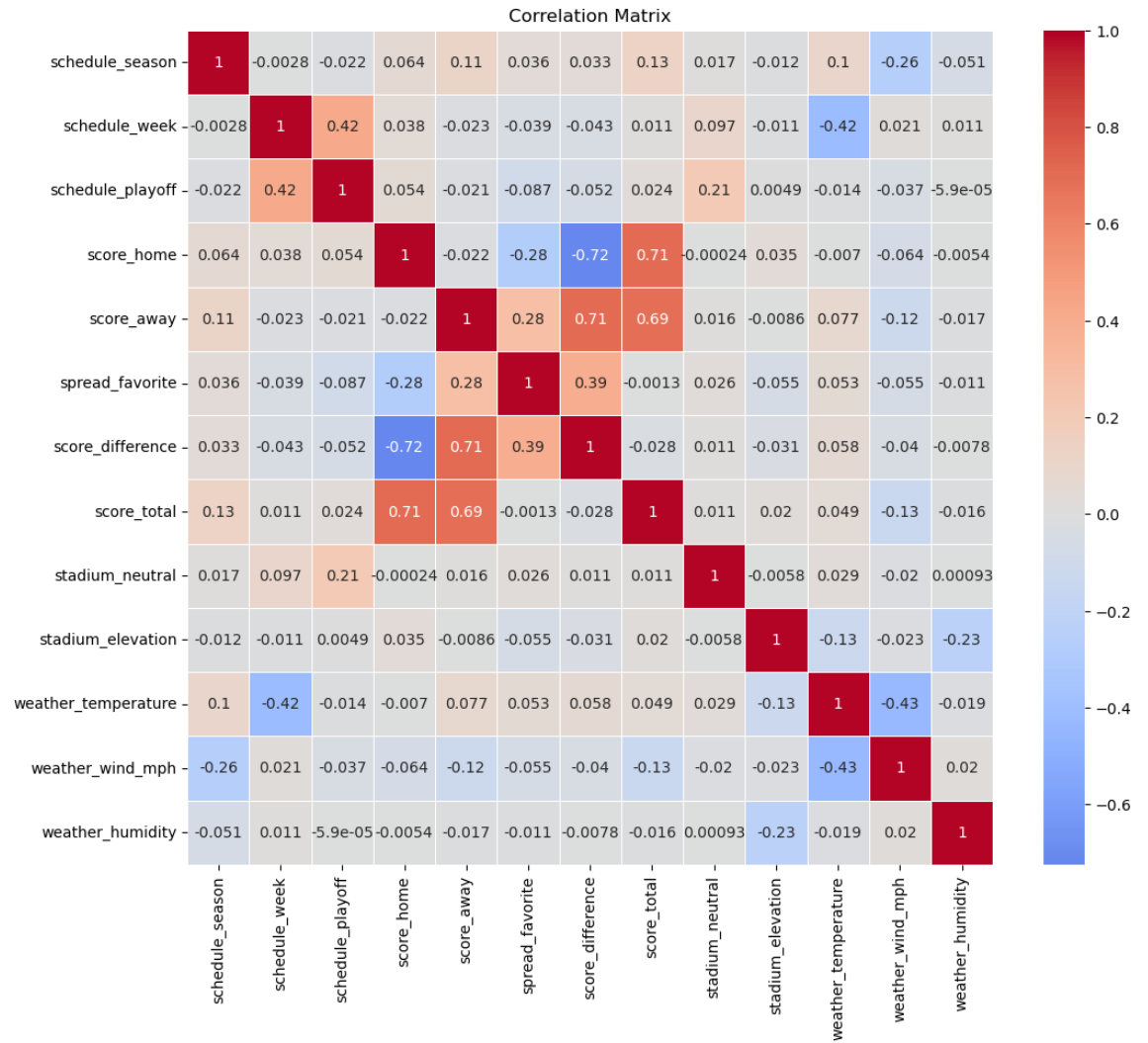


Fig. 9. Correlation of Variables

## 3 Model Creation

### 3.1 Pre-Processing

### 3.2 Model Selection and Parameter Tuning

### 3.3 Model Analysis

## 4 Conclusions

## References

1. <https://www.kaggle.com/docs/api>
2. Contessa Brewer, J.G.: A record 73 million americans plan to bet on the nfl this season, survey says (September 2003), <https://www.cnbc.com/2023/09/06/nfl-week-1-record-number-of-americans-to-bet-on-nfl-this-season-.html>
3. Preciado, D.: What is a spread in sports betting?, <https://www.forbes.com/betting/sports-betting/what-is-a-spread/#:~:text=If%20the%20spread%20is%20set,by%20more%20than%20seven%20points.>
4. Spreadspoke: Nfl scores and betting data, <https://www.kaggle.com/datasets/tobycrabtree/nfl-scores-and-betting-data/data>