CONVERSATIONAL AI

PROJECT

PRIYANSHI KHARBANDA

COE14
102103395

NAMAN BHATEJA

3Nc5
102115132

Data Collection Technique [notebook]

The data collection methodology employed in this project aims to provide a comprehensive analysis of Premier League team statistics over multiple seasons through web scraping on fbref.com. The process involves the following intricate steps:

1. Iterating Over Seasons

The script dynamically iterates over specified seasons, ensuring flexibility in data collection across different timeframes. This allows for a comprehensive historical analysis of team performance.
Python

```
for year in years:
```

2. Fetching Standings Table and Team URLs

An initial HTTP request is made to the Premier League statistics page on fbref.com, and the BeautifulSoup library is utilized to parse the HTML content. The standings table is isolated, and team URLs are extracted for subsequent data retrieval.

```
data = requests.get(standings_url)
soup = BeautifulSoup(data.text)
standings_table = soup.select('table.stats_table')[0]
links = [l.get("href") for l in standings_table.find_all('a')]
links = [l for l in links if '/squads/' in l]
team_urls = [f"https://fbref.com{l}" for l in links]
```

3. Fetching Match and Shooting Data for Each Team

The script further drills down into individual team statistics. It iterates through each team URL, fetching detailed match data and extracting shooting statistics for a more nuanced understanding of team dynamics.

```
for team_url in team_urls:
 # ... (code to fetch match and shooting data)
```

4. Merging Match and Shooting Data

To consolidate the obtained information, match and shooting data are merged based on the "Date" column. This step is crucial for creating a cohesive dataset for subsequent analysis.

```
team_data = matches.merge(shooting[["Date", "Sh", "SoT", "Dist",
"FK", "PK", "PKatt"]], on="Date")
```

5. Filtering Data for Premier League and Appending to Dataset

The collected data is meticulously filtered to include only Premier League matches, ensuring the integrity of the dataset. The information is then appended to the all_matches list, creating a comprehensive repository of team statistics.

```
team_data = team_data[team_data["Comp"] == "Premier League"]
team_data["Season"] = year
team_data["Team"] = team_name
all_matches.append(team_data)
```

6. Delay Between Requests

In acknowledgment of ethical web scraping practices, a deliberate 5-second pause is introduced between each request. This thoughtful approach mitigates the risk of overloading the server and demonstrates a commitment to responsible data collection.

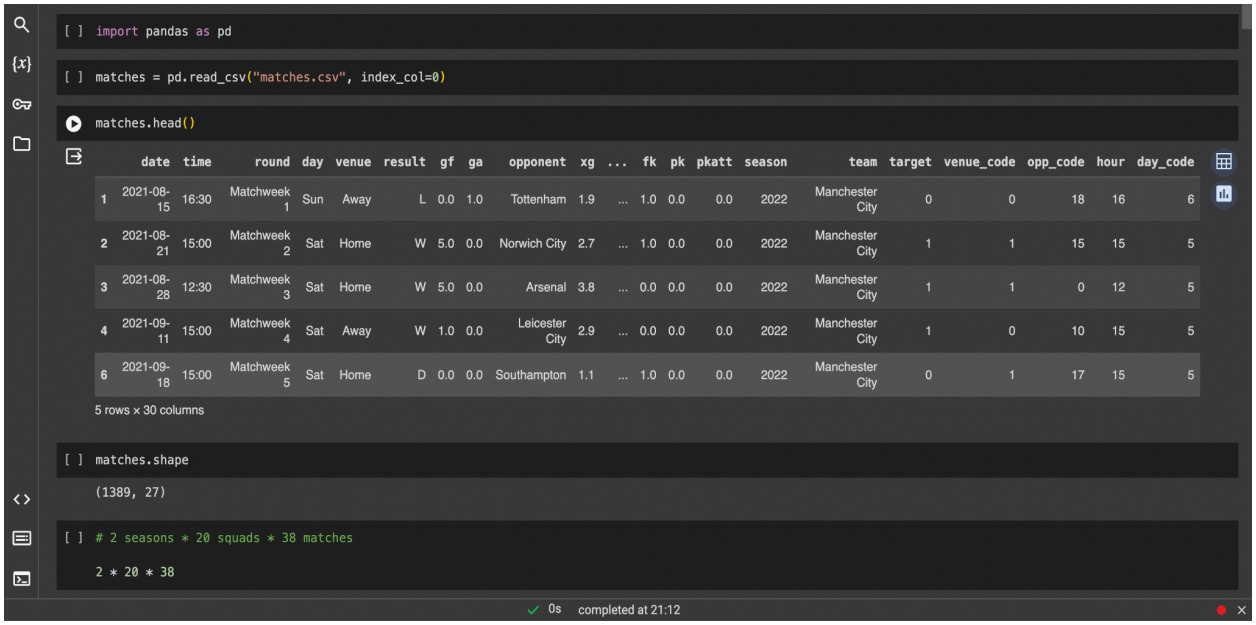
```
time.sleep(5)
```

Data Description [dataset link]

The resulting dataset is a rich and multifaceted compilation of statistical metrics pertaining to Premier League teams. Key performance indicators include:

- Goals (Sh): The total number of shots taken by a team.
- Shots on Target (SoT): The count of shots that successfully hit the target.
- Distance of Shots (Dist): The average distance of shots taken by the team, providing insights into attacking strategies.
- Free Kicks (FK): The frequency of free kicks taken, indicating set-piece proficiency.
- Penalties (PK): Total penalties scored, reflecting a team's ability to capitalize on key opportunities.
- Penalty Attempts (PKatt): The overall number of penalty attempts, showcasing aggression and drawing fouls.

This meticulously organized dataset is structured by season, team, and date, facilitating nuanced analyses of team dynamics over time. The conscientious inclusion of a time delay between requests underscores the commitment to ethical data acquisition.



```
[ ] import pandas as pd

[ ] matches = pd.read_csv("matches.csv", index_col=0)

matches.head()
```

	date	time	round	day	venue	result	gf	ga	opponent	xg	...	fk	pk	pkatt	season	team	target	venue_code	opp_code	hour	day_code
1	2021-08-15	16:30	Matchweek 1	Sun	Away	L	0.0	1.0	Tottenham	1.9	...	1.0	0.0	0.0	2022	Manchester City	0	0	18	16	6
2	2021-08-21	15:00	Matchweek 2	Sat	Home	W	5.0	0.0	Norwich City	2.7	...	1.0	0.0	0.0	2022	Manchester City	1	1	15	15	5
3	2021-08-28	12:30	Matchweek 3	Sat	Home	W	5.0	0.0	Arsenal	3.8	...	0.0	0.0	0.0	2022	Manchester City	1	1	0	12	5
4	2021-09-11	15:00	Matchweek 4	Sat	Away	W	1.0	0.0	Leicester City	2.9	...	0.0	0.0	0.0	2022	Manchester City	1	0	10	15	5
6	2021-09-18	15:00	Matchweek 5	Sat	Home	D	0.0	0.0	Southampton	1.1	...	1.0	0.0	0.0	2022	Manchester City	0	1	17	15	5

5 rows × 30 columns

```
[ ] matches.shape

(1389, 27)

[ ] # 2 seasons * 20 squads * 38 matches

2 * 20 * 38
```

0s completed at 21:12

Data Preprocessing [notebook]

Data Preprocessing: Feature Engineering for Match Analysis

The following feature engineering techniques have been employed to make the data more suitable for deploying random forest model as it takes categorical data.

1. Day Code Assignment

To provide a numerical representation of the day of the week for each match, a new column named "day_code" has been introduced. This column assigns a specific code to each day, allowing for potential insights into whether certain teams perform differently on specific days.

```
matches["day_code"] = matches["date"].dt.dayofweek
```

2. Hour Extraction for Time Analysis

Aiming to explore potential correlations between match performance and the time of day, the "hour" column has been created. Extracting the hour from the "time" column and converting it into an integer facilitates a detailed examination of whether particular time slots influence team performance.

```
matches["hour"] = matches["time"].str.replace(":.+", "", regex=True).astype("int")
```

3. Opponent Encoding

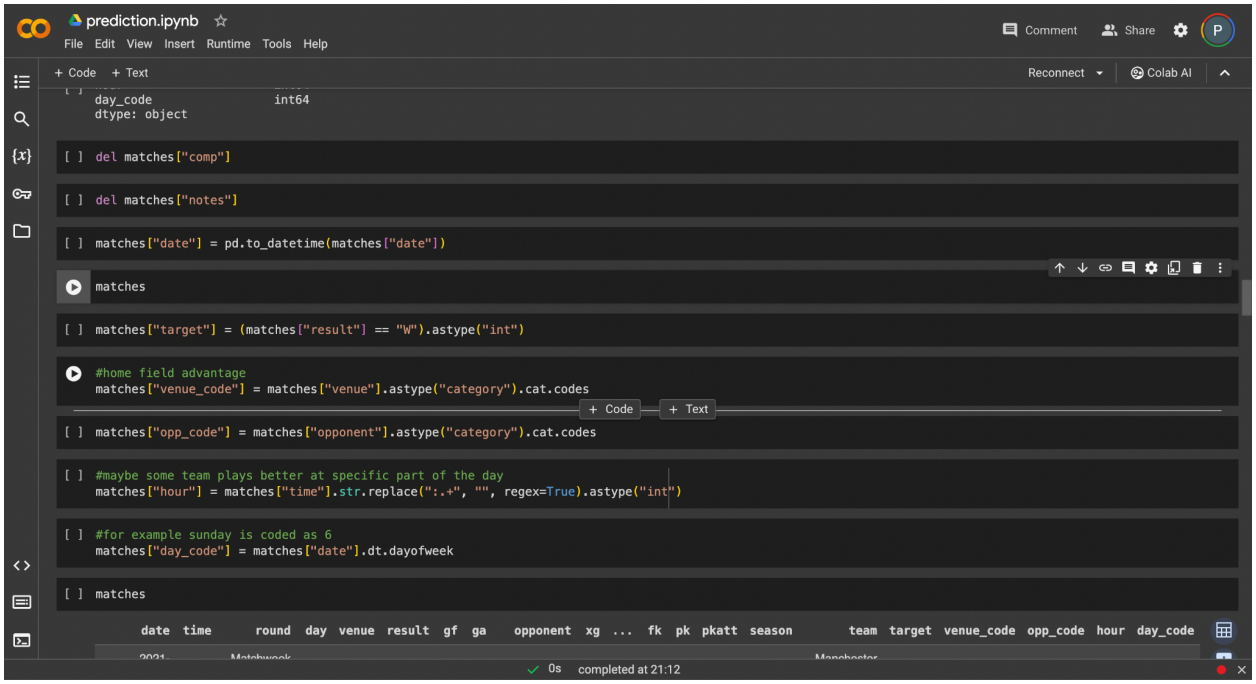
In order to quantify opponent information, the "opp_code" column is introduced. Opponents are encoded as numerical values, providing a structured representation for further analysis of team performance against specific rivals.

```
matches["opp_code"] = matches["opponent"].astype("category").cat.codes
```

4. Venue Encoding for Home Field Advantage

Recognizing the potential influence of the playing venue on team performance, the "venue_code" column has been created. This column encodes the venue information, with a specific focus on capturing the notion of home field advantage.

```
matches["venue_code"] = matches["venue"].astype("category").cat.codes
```



Data Preprocessing: Model Evaluation

The football match data is loaded into a DataFrame, and the 'date' column is used to split the dataset into training and testing sets based on a specified date ('2022-01-01').

```
train = matches[matches["date"] < '2022-01-01']
test = matches[matches["date"] > '2022-01-01']
```

1. Feature Selection:

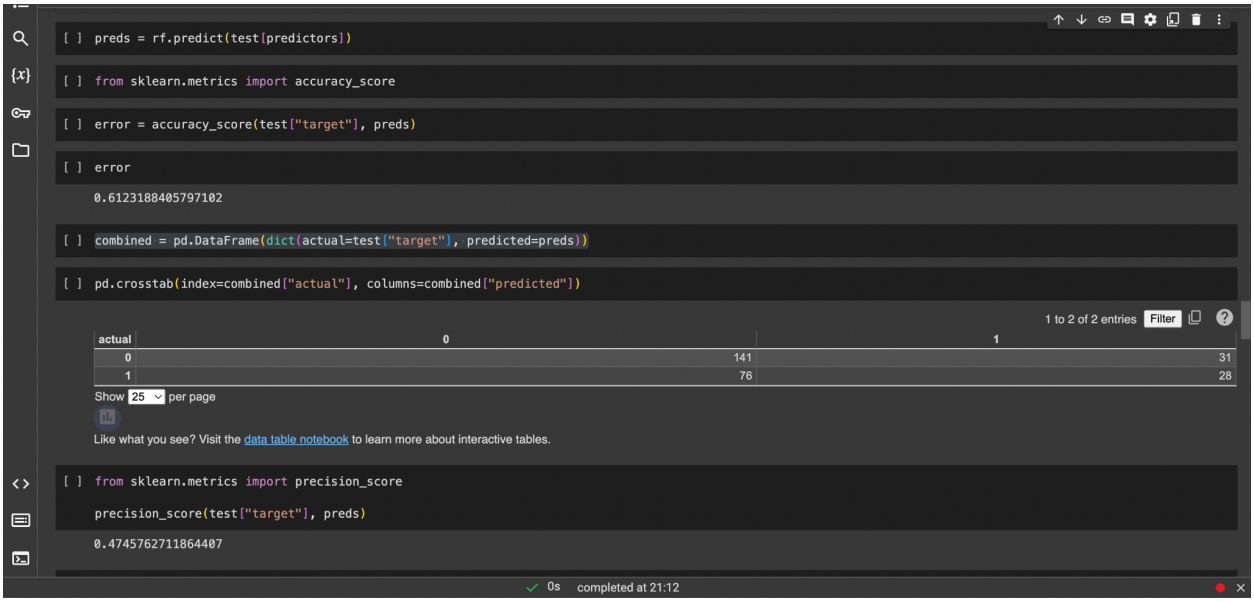
The predictor variables for the model are chosen as ["venue_code", "opp_code", "hour", “day_code”

```
predictors = ["venue_code", "opp_code", "hour", "day_code"]
rf.fit(train[predictors], train["target"])
```

2. Model Evaluation:

The model is evaluated using accuracy_score and precision_score metrics on the test set.

```
preds = rf.predict(test[predictors])
error = accuracy_score(test["target"], preds)
precision = precision_score(test["target"], preds)
```



3. Rolling Averages:

Rolling averages have been computed for key performance metrics, such as goals for (gf), goals against (ga), shots (sh), shots on target (sot), distance (dist), free kicks (fk), penalties (pk), and penalty attempts (pkatt). These rolling averages are applied to the data grouped by team, providing smoothed representations of team performance over time using a custom function 'rolling_averages'. The rolling window is set to 3 matches.

```
cols = ["gf", "ga", "sh", "sot", "dist", "fk", "pk", "pkatt"]
new_cols = [f"{c}_rolling" for c in cols]
rolling_averages(group, cols, new_cols)
```

4. Applying Rolling Averages:

Rolling averages are applied to the entire dataset by grouping it by team and using the 'rolling_averages' function.

```
matches_rolling = matches.groupby("team").apply(lambda x: rolling_averages(x, cols, new_cols))
matches_rolling = matches_rolling.droplevel('team')
matches_rolling.index = range(matches_rolling.shape[0])
```

5. Model Training with Rolling Averages:

The RandomForestClassifier is re-trained using the original predictors along with the newly created rolling averages features.

```
combined, precision = make_predictions(matches_rolling, predictors + new_cols)
```

6. Result Integration:

The results of the model predictions are integrated back into the dataset for further analysis.

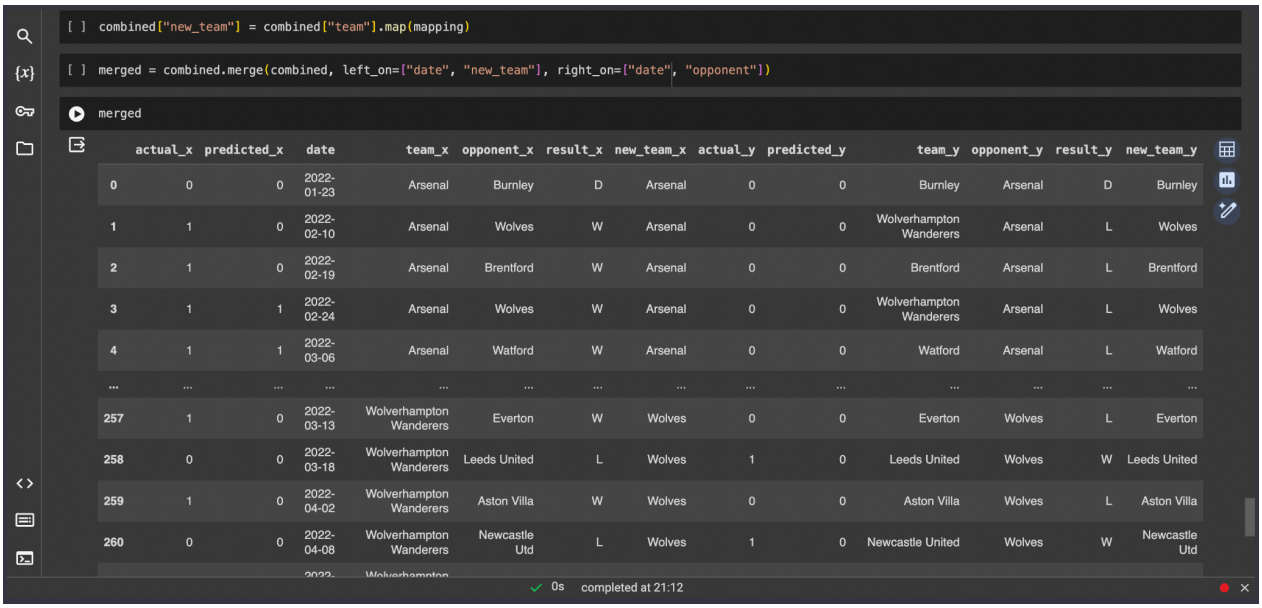
```
combined = combined.merge(matches_rolling[["date", "team", "opponent", "result"]],
left_index=True, right_index=True)
```

7. Team Name Mapping:

A custom dictionary is used to map certain team names to their shorter versions for better readability.

```
map_values = {"Brighton and Hove Albion": "Brighton", "Manchester United":
"Manchester Utd", ...}
mapping = MissingDict
(**map_values)
```

End results

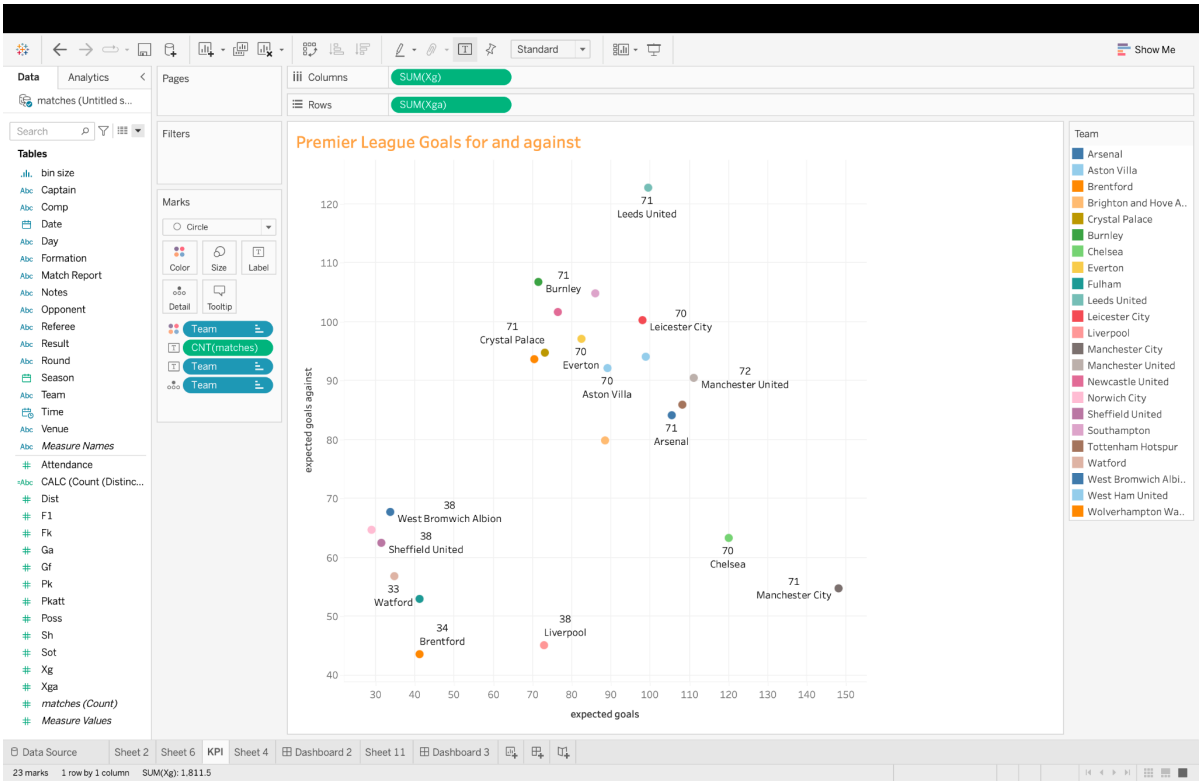


```
[ ] combined["new_team"] = combined["team"].map(mapping)

[ ] merged = combined.merge(combined, left_on=["date", "new_team"], right_on=["date", "opponent"])

merged
```

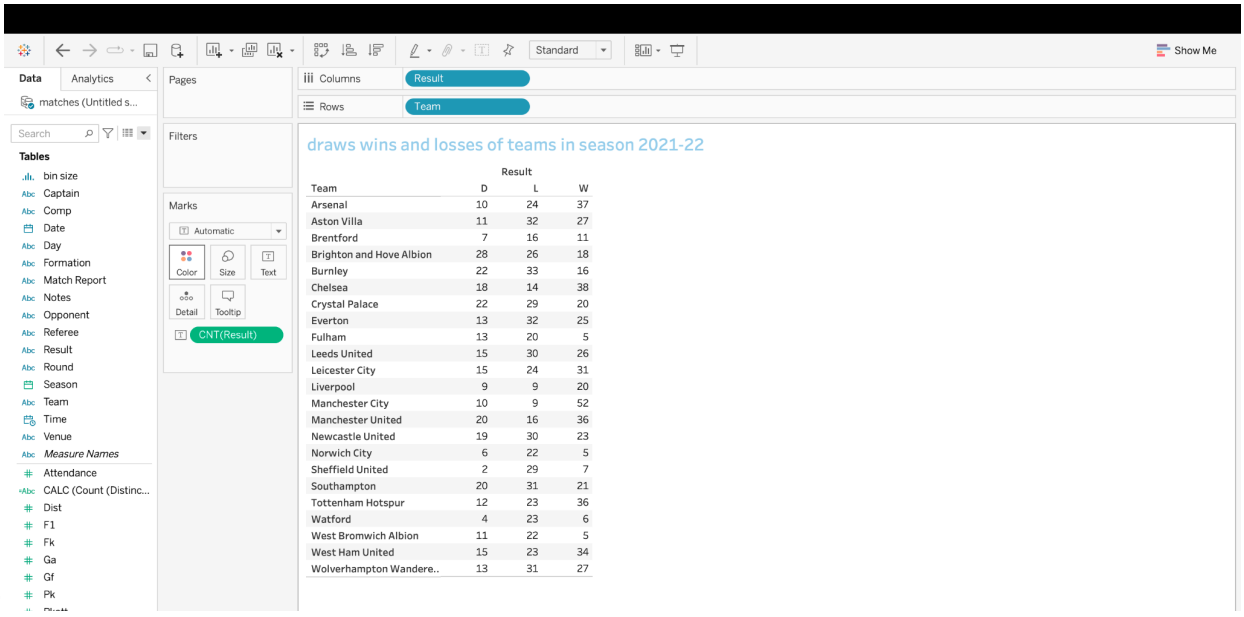
	actual_x	predicted_x	date	team_x	opponent_x	result_x	new_team_x	actual_y	predicted_y	team_y	opponent_y	result_y	new_team_y
0	0	0	2022-01-23	Arsenal	Burnley	D	Arsenal	0	0	Burnley	Arsenal	D	Burnley
1	1	0	2022-02-10	Arsenal	Wolves	W	Arsenal	0	0	Wolverhampton Wanderers	Arsenal	L	Wolves
2	1	0	2022-02-19	Arsenal	Brentford	W	Arsenal	0	0	Brentford	Arsenal	L	Brentford
3	1	1	2022-02-24	Arsenal	Wolves	W	Arsenal	0	0	Wolverhampton Wanderers	Arsenal	L	Wolves
4	1	1	2022-03-06	Arsenal	Watford	W	Arsenal	0	0	Watford	Arsenal	L	Watford
...	...	...	...	...	...	...	...	...	...	...	...	...	...
257	1	0	2022-03-13	Wolverhampton Wanderers	Everton	W	Wolves	0	0	Everton	Wolves	L	Everton
258	0	0	2022-03-18	Wolverhampton Wanderers	Leeds United	L	Wolves	1	0	Leeds United	Wolves	W	Leeds United
259	1	0	2022-04-02	Wolverhampton Wanderers	Aston Villa	W	Wolves	0	0	Aston Villa	Wolves	L	Aston Villa
260	0	0	2022-04-08	Wolverhampton Wanderers	Newcastle Utd	L	Wolves	1	0	Newcastle United	Wolves	W	Newcastle Utd

Dashboard creation and worksheet snapshots {link}

Worksheet snapshots

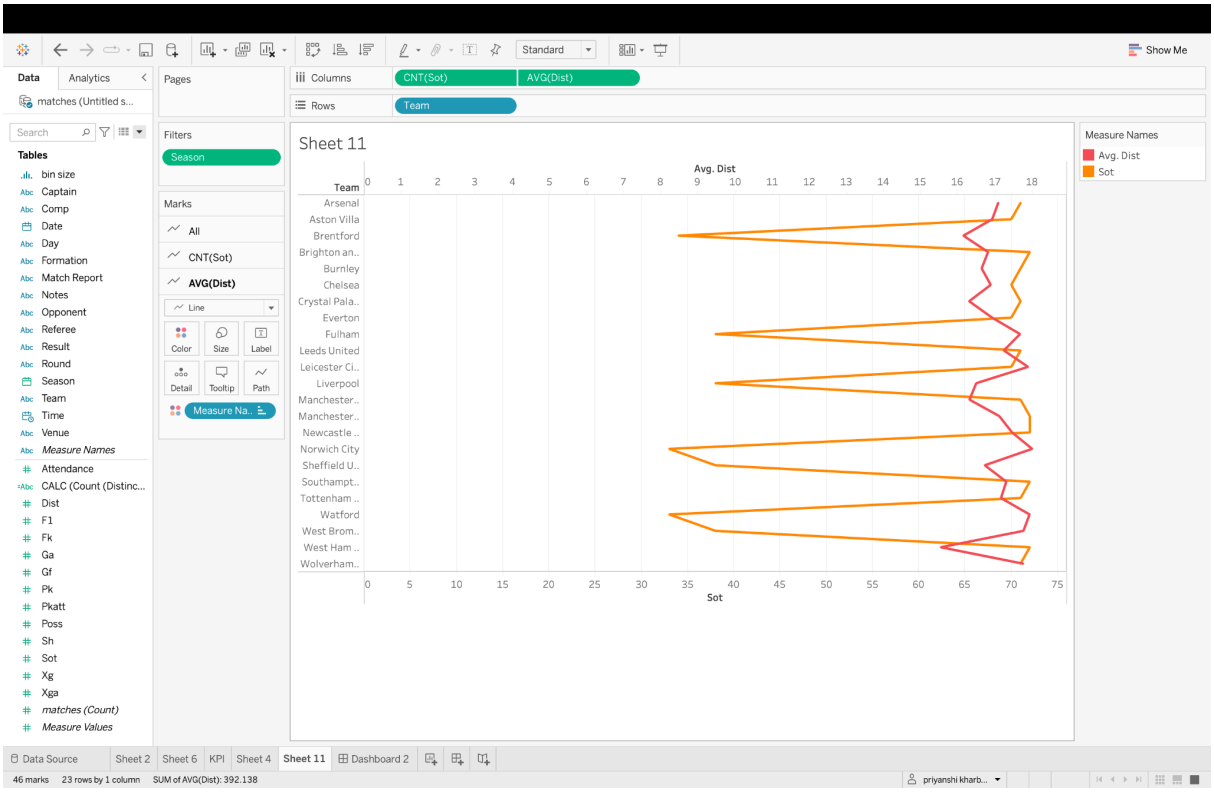


1

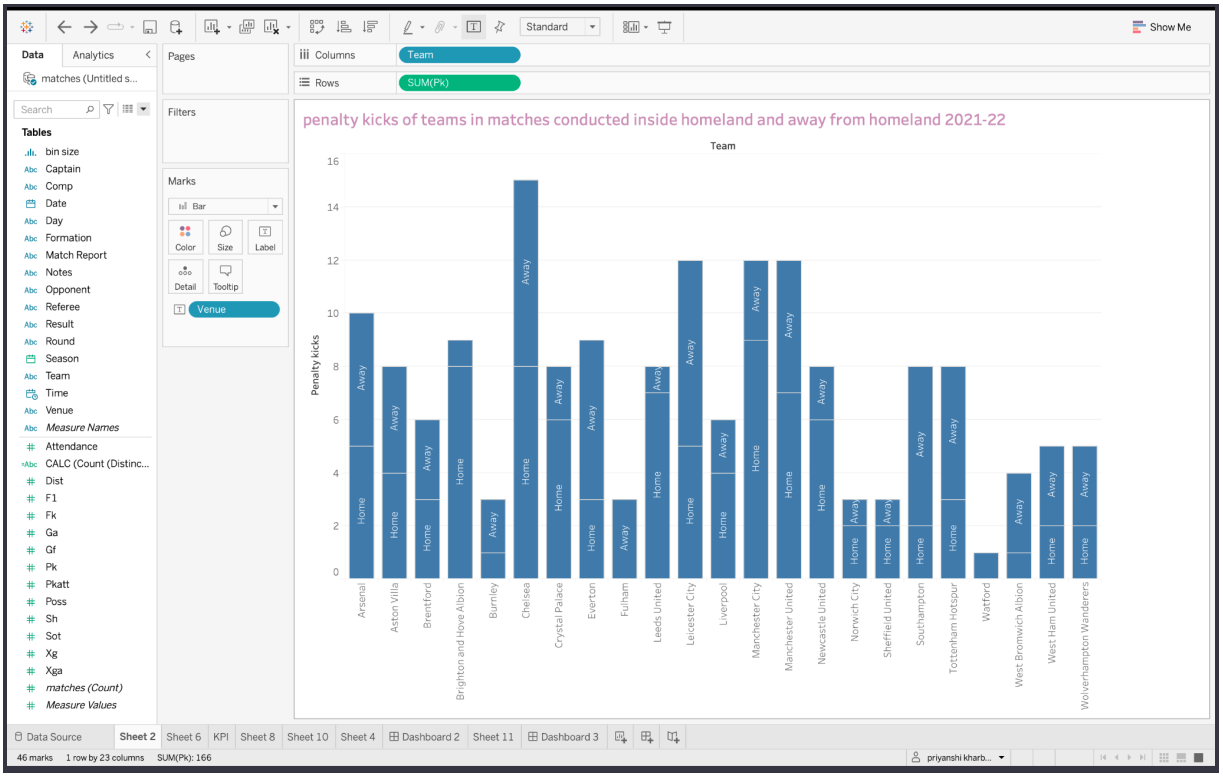


2

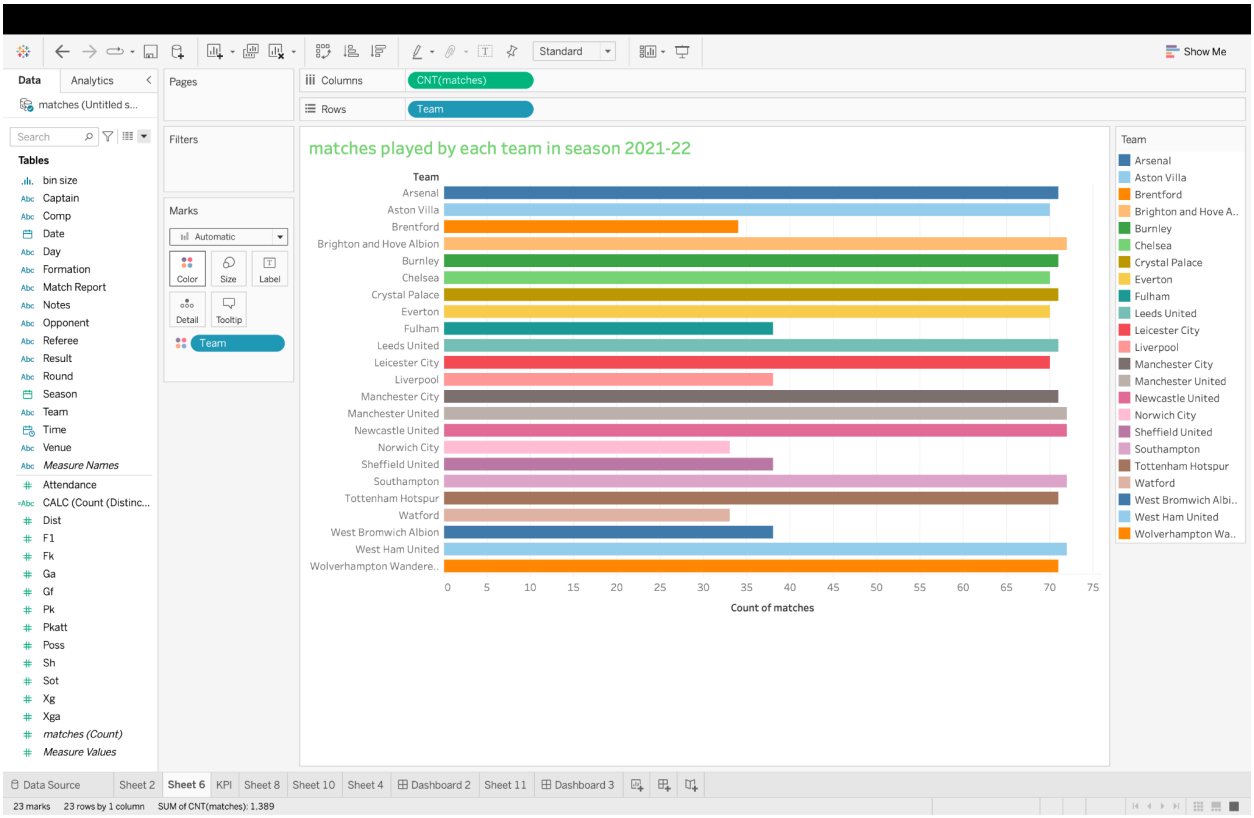
[1] Goals for and against by each team
[2] draws win and losses in a season by each team.



Line chart between the avg shot distance of team and it's shot on target



Penalty kicks of temas in matches conducted inside homeland and away from homeland



Matches played by each team in season of 21-22