BRAZILIAN PLAYERS PERFORMANCE IN THE CHAMPIONS LEAGUE
AND MARKET VALUE ANALYSIS

FINAL PROJECT

PASSION
FOR FOOTBALL

* * *

GOALS
OF THE ANALYIS

01

IDENTIFY PLAYER PERFORMANCE AND HOW ITS IMPACTS ON MATCH OUTCOMES

02

ANALYZE AND PREDICT PLAYERS' MARKET VALUE

DATASET AND METHODS

“FOOTBALL DATA” FROM KAGGLE

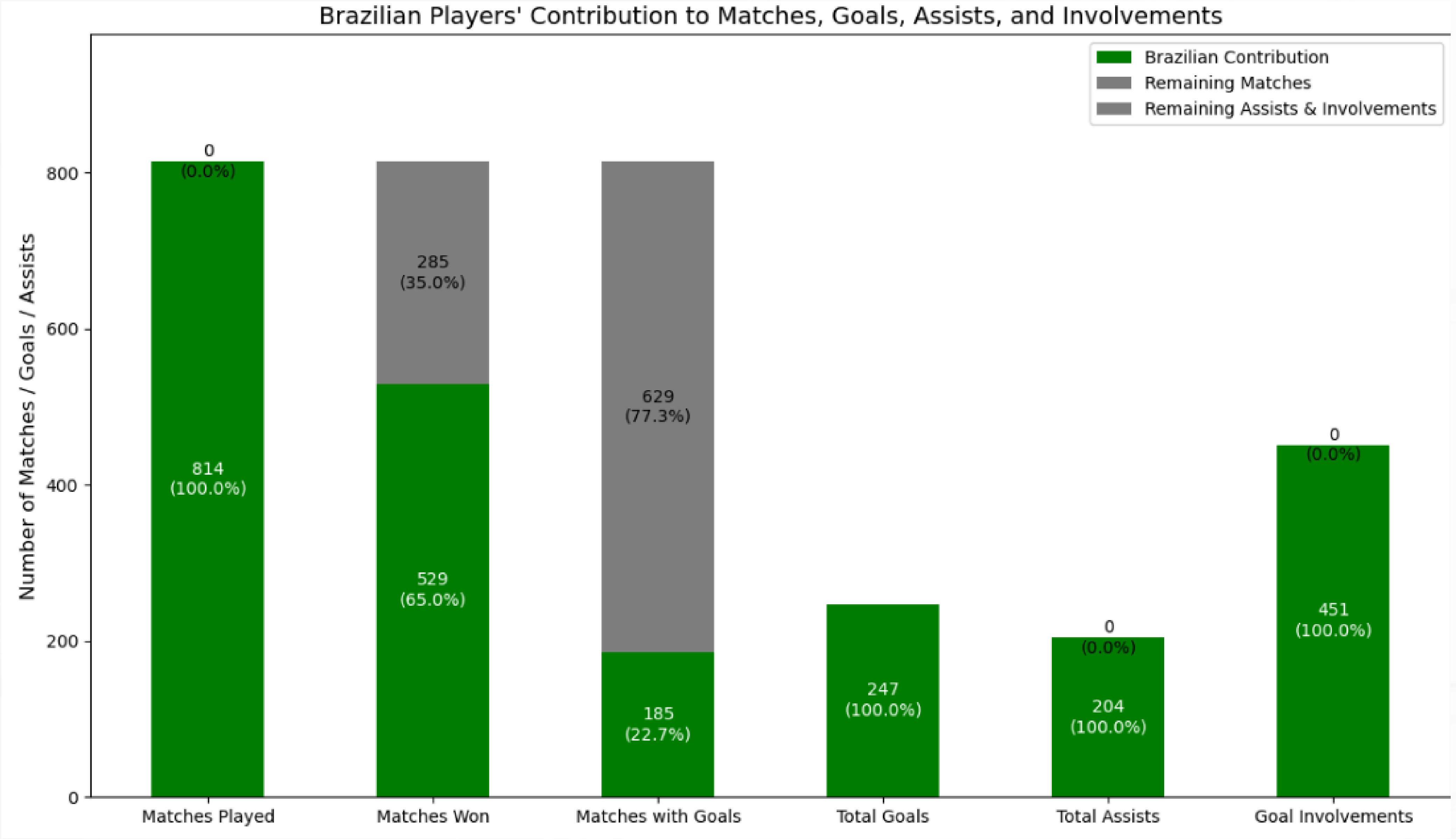
DATA CLEANING AND PREPROCESSING

EXPLORATORY ANALYSIS AND VISUALIZATIONS

MACHINE LEARNING

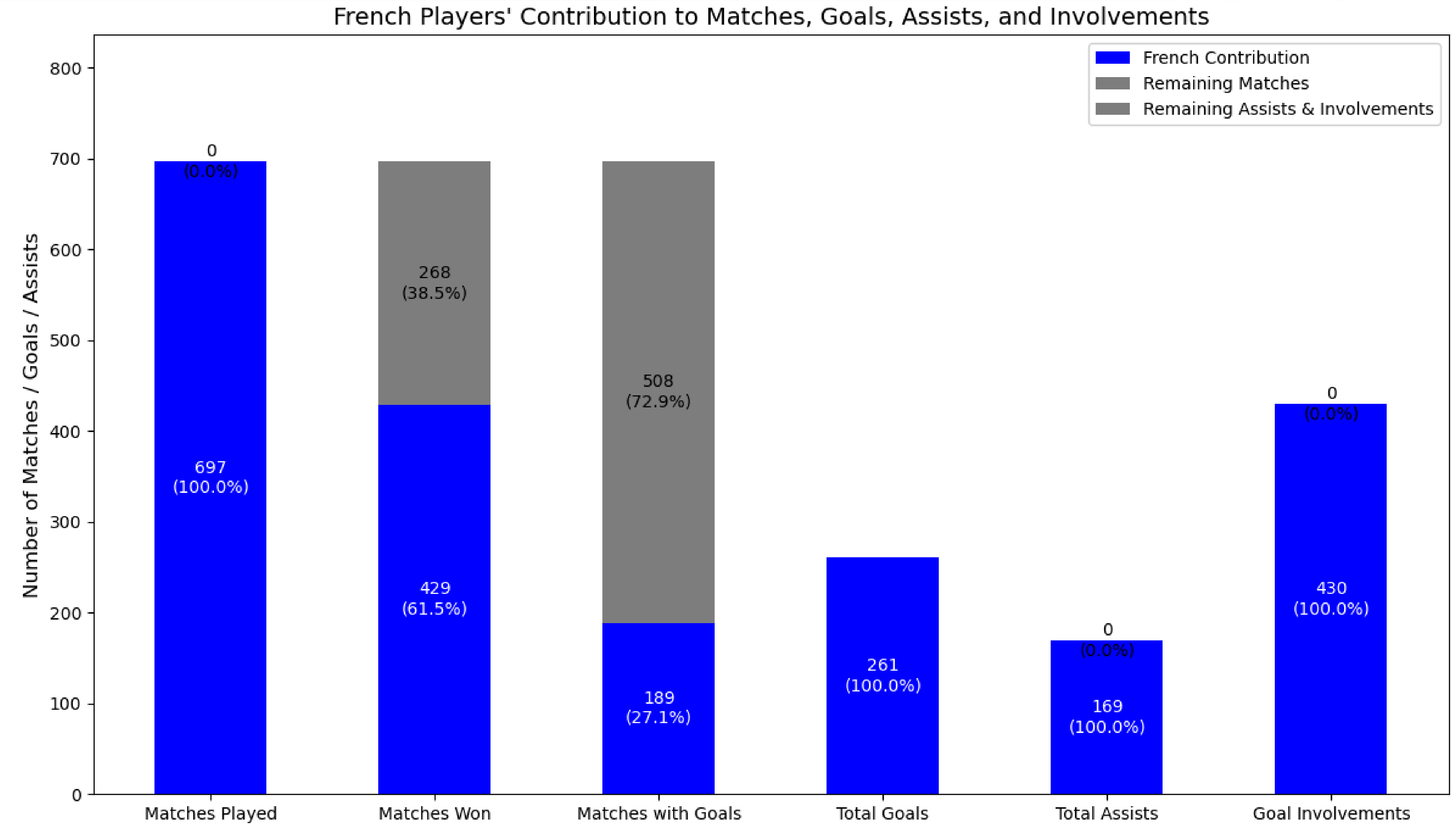
BRAZILIAN PLAYERS PERFORMANCES

THE IMPACT OF BRAZILIAN PLAYERS IN THE CHAMPIONS LEAGUE



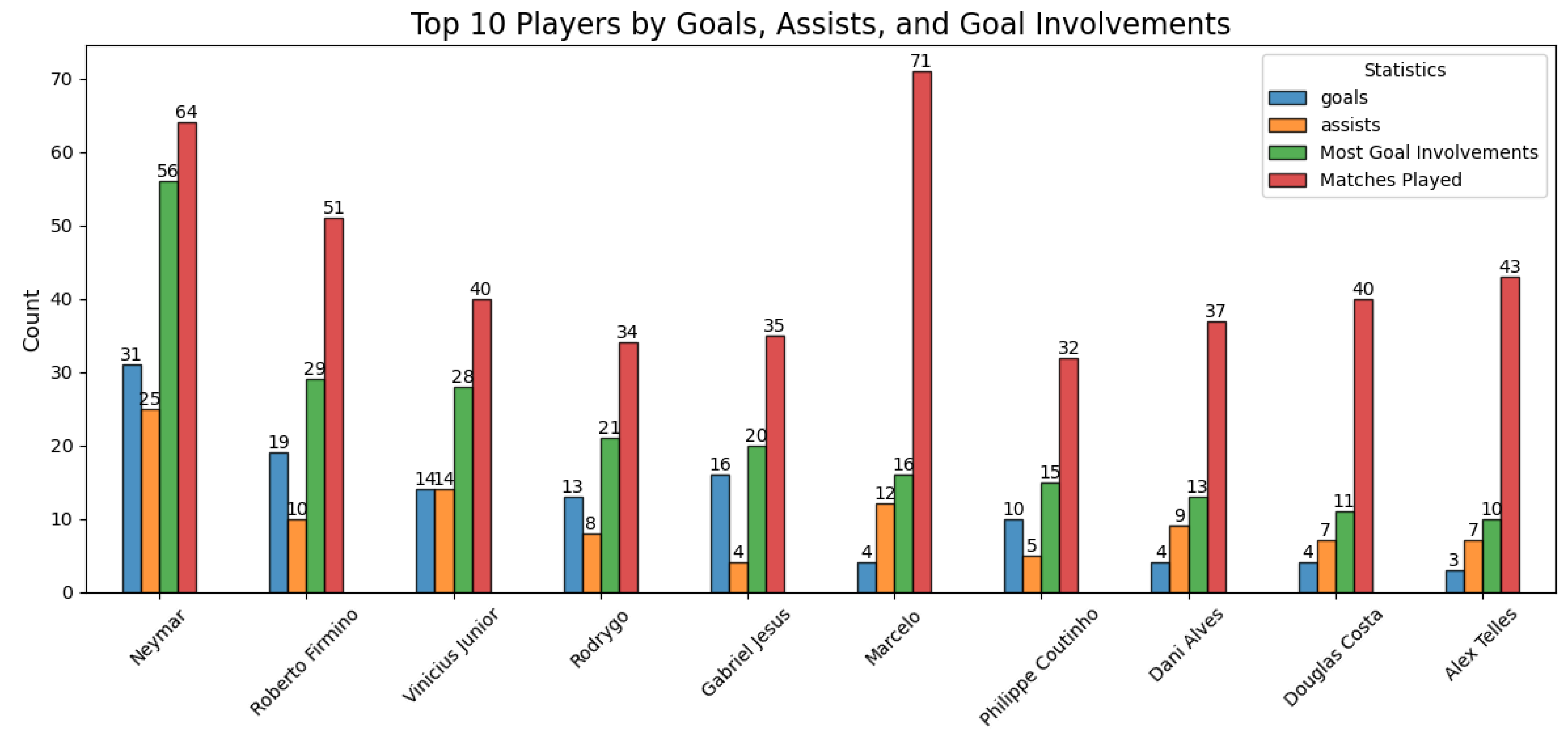
FRENCH PLAYERS PERFORMANCES

THE IMPACT OF FRENCH PLAYERS IN THE CHAMPIONS LEAGUE



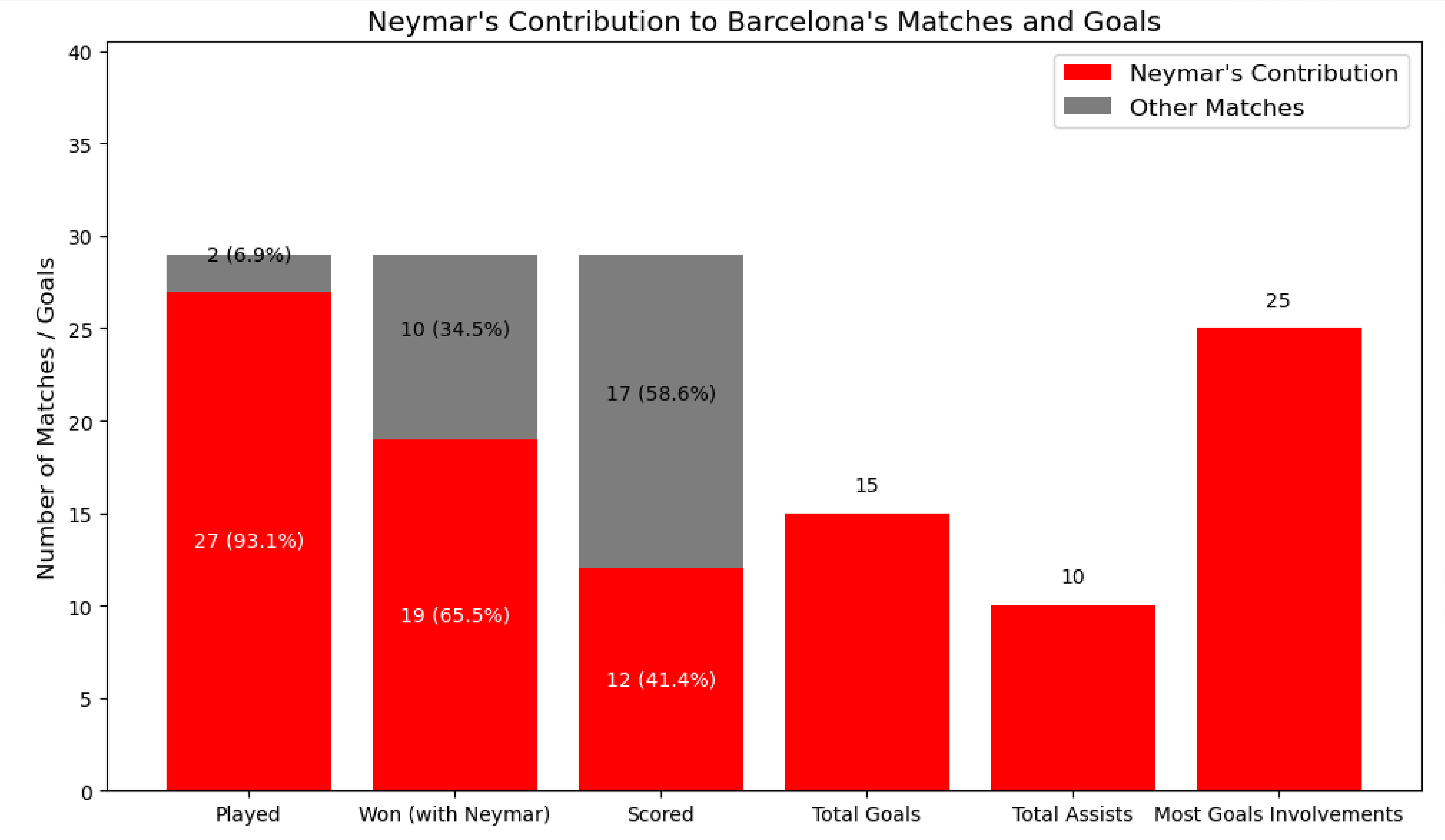
TOP 10 BRAZILIAN PLAYERS

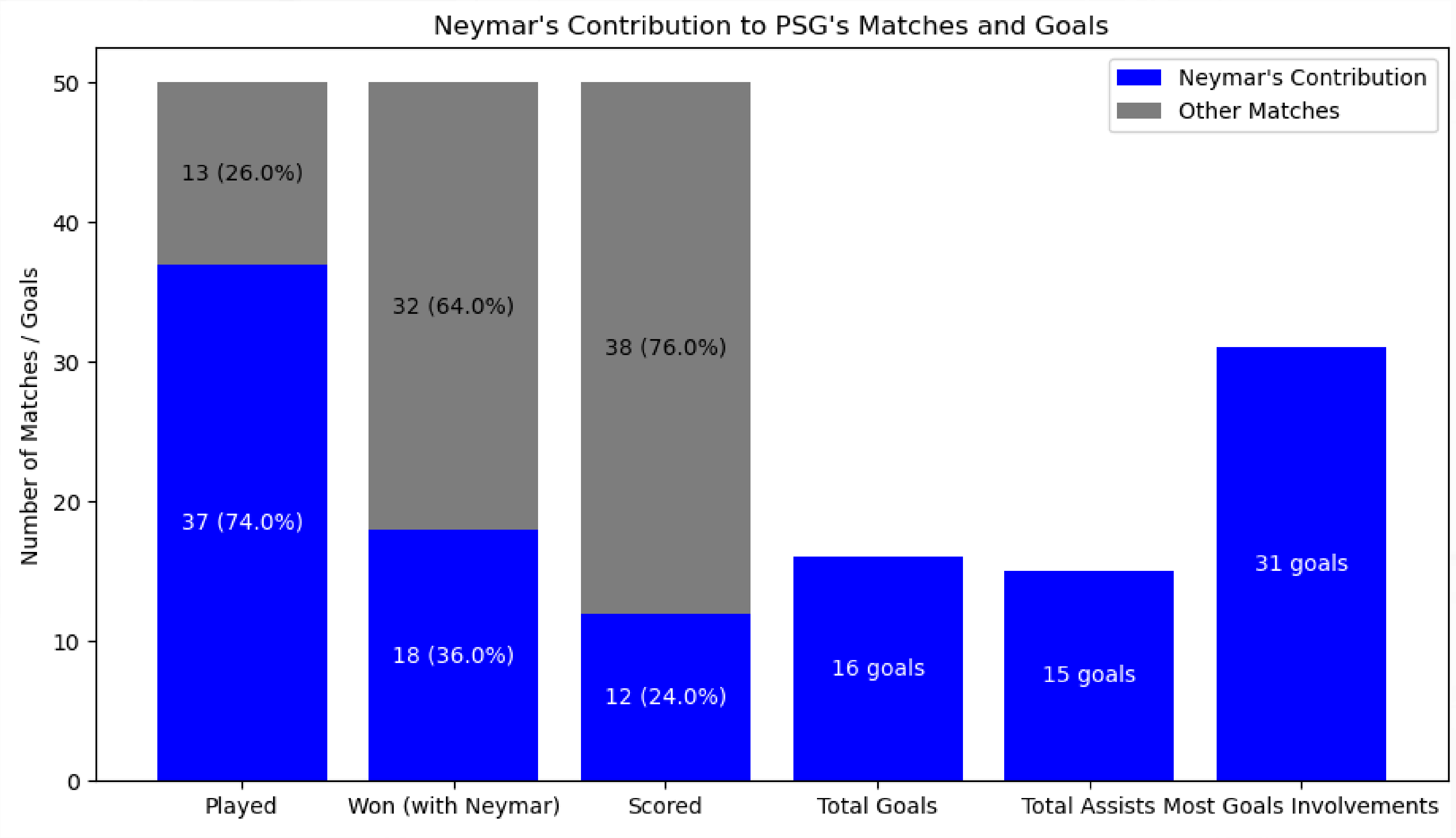
THE IMPACT OF TOP 10 BRAZILIAN PLAYERS IN THE CHAMPIONS LEAGUE MATCHES



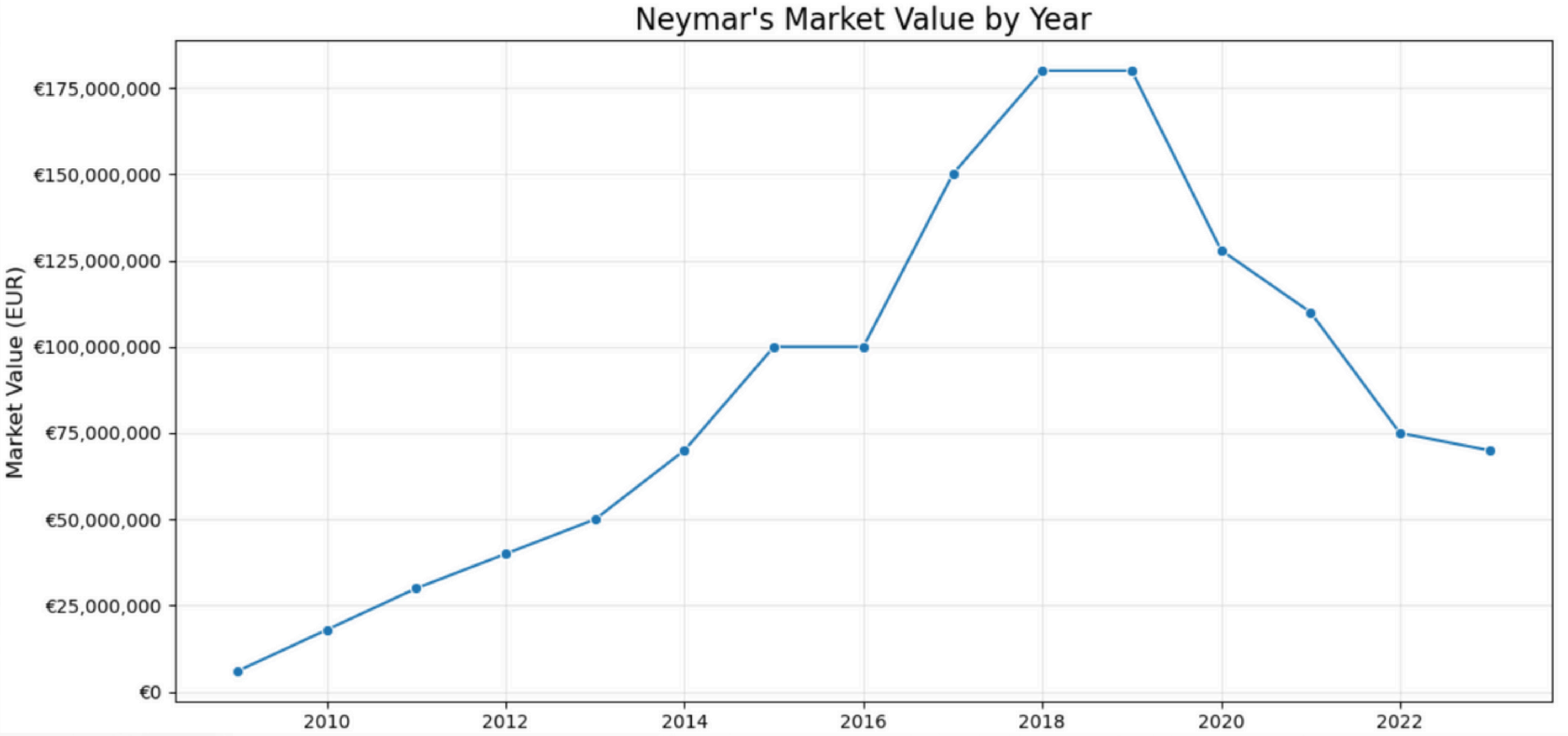
BEST BRAZILIAN PLAYER

THE IMPACT OF NEYMAR IN CHAMPIONS LEAGUE MATCHES



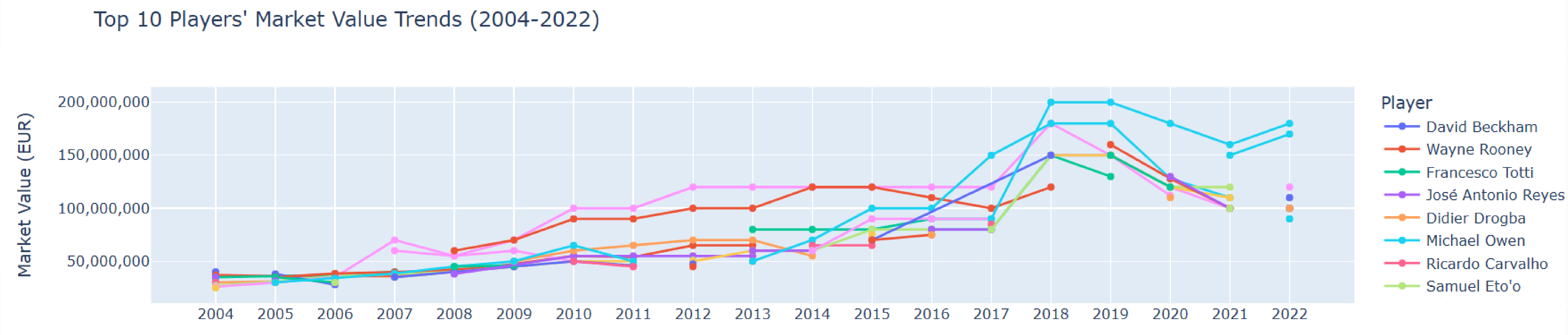


NEYMAR MARKET VALUE



SEASONAL TREND NEYMAR MARKET VALUE

MARKET VALUE ANALYSIS AND MACHINE LEARNING PREDICTION



DATA PREPROCESSING AND DATA CLEANING

Data Preprocessing

```
players_df_final = players_df_final.drop(['date_of_birth', 'foot', 'market_value_in_eur', 'date', 'dateweek', 'year', 'player_name', 'competition_id', 'last_season'], axis=1)
players_df_final = players_df_final.rename(columns={'Total Most Goal Involvements': 'Total_Most_Goal_Involvements'})

age_mean = players_df_final['Age'].mean()
Total_Most_Goal_Involvements_median = players_df_final['Total_Most_Goal_Involvements'].median()

players_df_final.fillna({'Age': age_mean}, inplace=True)
players_df_final.fillna({'Total_Most_Goal_Involvements': Total_Most_Goal_Involvements_median}, inplace=True)
players_df_final = players_df_final.rename(columns={'name': 'League'})

league_name_mapping = {
    'serie-a': 'Serie A',
    'bundesliga': 'Bundesliga',
    'super-league-1': 'Super League 1',
    'premier-league': 'Premier League',
    'super-lig': 'Super Lig',
    'premier-liga': 'Primeira Liga',
    'laliga': 'La Liga',
    'superligaen': 'Superligaen',
    'scottish-premiership': 'Scottish Premiership',
    'eredivisie': 'Eredivisie',
    'ligue-1': 'Ligue 1',
    'jupiler-pro-league': 'Jupiler Pro League',
    'liga-portugal-bwin': 'Liga Portugal Bwin'
}
players_df_final['League'] = players_df_final['League'].replace(league_name_mapping)
```

ENCODING
DATA SPLITTING AND TRAIN/TEST

Encoding

```
players_df_final = players_df_final.drop(['club_name', 'player_id', 'country_name', 'League', 'position'], axis=1)

# Frequency Encoding
nationality_counts = players_df_final['nationality'].value_counts()
players_df_final['nationality_encoded'] = players_df_final['nationality'].map(nationality_counts)
players_df_final = players_df_final.drop(columns=['nationality'])

ordinal_encoder = OrdinalEncoder()
players_df_final[['league_prestige', 'club_prestige']] = ordinal_encoder.fit_transform(players_df_final[['league_prestige', 'club_prestige']])
```

```
# Split the data

y = players_df_final['market_value_in_eur_x']
X = players_df_final.drop(columns=['market_value_in_eur_x'])
```

```
# Train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

ROBUST SCALER LINEAR REGRESSION

FIRST MODEL EVALUATION

```
# Train Linear Regression Model
lr_model = LinearRegression()
lr_model.fit(X_train_scaled, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
# Predictions
y_pred_lr = lr_model.predict(X_test_scaled)
```

```
# Model Evaluation
print("Linear Regression Performance:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_lr)}")
print(f"RMSE: {mean_squared_error(y_test, y_pred_lr, squared=False)}")
print(f"R² Score: {r2_score(y_test, y_pred_lr)}")
```

```
Linear Regression Performance:
MAE: 482966.00146758836
RMSE: 1609493.8302030105
R² Score: 0.12250691550182824
```

```
# Feature Scaling
scaler = RobustScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

RANDOM FOREST REGRESSION

SECOND MODEL EVALUATION

```
# Train Random Forest Model
rf_model = RandomForestRegressor(n_estimators=100, max_features=3, max_samples=None, oob_score=True, random_state=42)
rf_model.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor(max_features=3, oob_score=True, random_state=42)
```

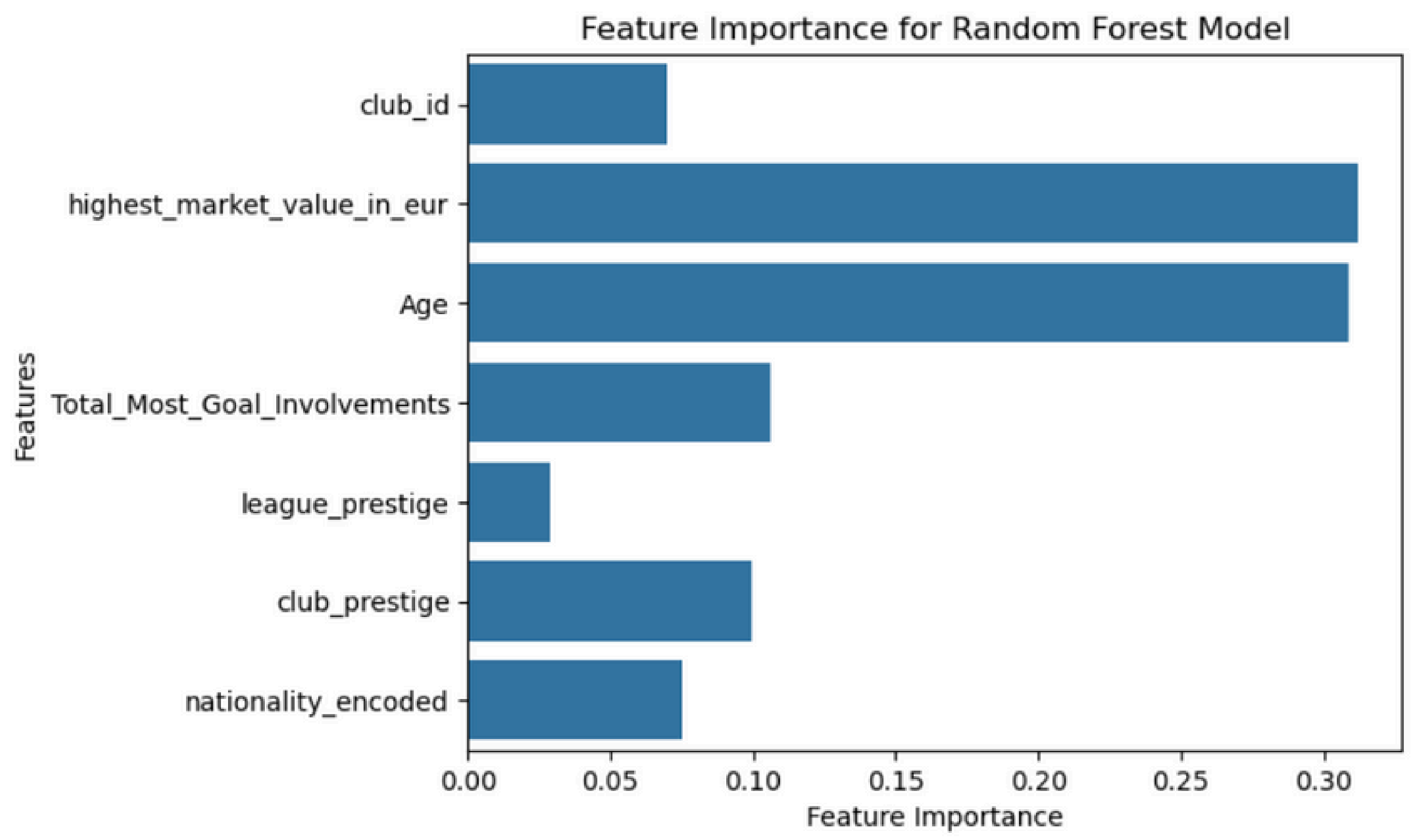
```
# Predictions
y_pred_rf = rf_model.predict(X_test)
```

```
# Random Forest Evaluation
print("\nRandom Forest Performance:")
print(f"MAE: {mean_absolute_error(y_test, y_pred_rf)}")
print(f"RMSE: {mean_squared_error(y_test, y_pred_rf, squared=False)}")
print(f"R² Score: {r2_score(y_test, y_pred_rf)}")
print(f"OOB Score: {rf_model.oob_score_}")
```

```
Random Forest Performance:
MAE: 328570.26483807113
RMSE: 1267362.0597596057
R² Score: 0.45591534989597504
OOB Score: 0.49459716833429257
```

FEATURE IMPORTANCE
FOR RANDOM FOREST REGRESSION

```python
# Feature Importance Plot
sns.barplot(x=rf_model.feature_importances_, y=X_train.columns)
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("Feature Importance for Random Forest Model")
plt.show()
```



ACTUAL MARKET VALUES
PREDICTED VALUES

	Actual Market Value (EUR)	Predicted Market Value (Linear Regression)	Predicted Market Value (Random Forest)
0	€50,000	€243,046	€189,750
1	€150,000	€407,836	€121,400
2	€200,000	€657,483	€402,950
3	€600,000	€-4,017	€155,000
4	€50,000	€13,840	€123,750
...	...	...	...
3419	€250,000	€368,097	€85,250
3420	€100,000	€657,568	€363,250
3421	€700,000	€513,400	€423,000

THANK YOU!