Predicted_Manager_Points

March 1, 2025

[1]:
```
import requests
import pandas as pd
from bs4 import BeautifulSoup

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import NoSuchElementException
from selenium.common.exceptions import TimeoutException
from selenium.common.exceptions import ElementClickInterceptedException
from selenium.webdriver.common.action_chains import ActionChains
import undetected_chromedriver as uc
import time
from fractions import Fraction
from collections import defaultdict
from unicodedata import normalize
```

This code scrapes several betting odds from Oddschecker.com, converts the odds to percentages and calculates predicted points for each Manager of next full gameweek in Fantasy Premier League according to the percentages. In addition to selenium, webdriver has to be installed also. Webdrivers run or drive a browser from inside of your code. Version of webdriver has to match the version of your browser.

Please see below for how managers will score points for your FPL team.

Metric	Points	Description
Win	+6	A victory in the game
Draw	+3	A tied game
Loss	0	No points for a defeat
Goal	+1	Each goal scored
Clean Sheet	+2	No goals conceded
Table Bonus (Draw)	+5	Extra points for drawing against a stronger team
Table Bonus (Win)	+10	Extra points for winning against a stronger team

The Table Bonus are the extra points that a manager receives when either winning or drawing against a team that is FIVE places above them in the table at the start of the Gameweek. For example, if a team is in 20th place, they will be eligible for the bonus against a team in 15th position or above. If the manager is eligible for the Table Bonus at the beginning of the Gameweek, they'll still be eligible regardless of the changes to league positions over the course of the Gameweek. Updated league positions will only be considered at the beginning of the NEXT Gameweek.

[2]:
```
url = "https://fantasy.premierleague.com/api/fixtures/"
response = requests.get(url)
if response.status_code != 200:
    raise Exception(f"Failed to fetch fixtures: {response.status_code}")
fixtures = response.json()
```

[3]:
```
game_weeks = defaultdict(list)
for fixture in fixtures:
    game_weeks[fixture["event"]].append(fixture)
for event in sorted(game_weeks.keys()):
    if all(not fixture['finished_provisional'] for fixture in␣
 ↪game_weeks[event]):
        next_gameweek = event
        break
    else:
        next_gameweek = None
```

[4]:
```
url = "https://fantasy.premierleague.com/api/bootstrap-static/"
response = requests.get(url)
if response.status_code != 200:
    raise Exception(f"Failed to fetch teams: {response.status_code}")
data = response.json()
teams = data['teams']
```

[5]:
```
next_gw_fixtures = [fixture for fixture in fixtures if fixture['event'] ==␣
↪next_gameweek]
```

[6]:
```
TEAM_NAMES_ODDSCHECKER = {
    "Nott'm Forest": "Nottingham Forest",
    "Wolves": "Wolverhampton",
    "Spurs": "Tottenham",
    }
```

[14]:
```
team_id_to_name = {team['id']: team['name'] for team in teams}
team_position = {team['id']: team['position'] for team in teams}
data = []
driver = uc.Chrome()  # Replace with the path to your WebDriver if needed
driver.get("https://www.oddschecker.com/football/english/premier-league/")

wait = WebDriverWait(driver, 10)

try:
    span_element = wait.until(EC.element_to_be_clickable((By.XPATH, '/html/body/
↪div[1]/div/section/h2/span[2]')))
    # Click on the <span> element
    span_element.click()

except TimeoutException:
    print()

wait = WebDriverWait(driver, 10)
try:
    cookiebutton = wait.until(EC.element_to_be_clickable((By.CLASS_NAME,␣
↪'CookieBannerAcceptButton_c1mxe743')))
    cookiebutton.click()
except TimeoutException:
    print()

except ElementClickInterceptedException:
    try:
        wait = WebDriverWait(driver, 10)
        cookiebutton = wait.until(EC.element_to_be_clickable((By.CLASS_NAME,␣
↪'CookieBannerAcceptButton_c1mxe743')))
        cookiebutton.click()
    except ElementClickInterceptedException:
        print()

for fixture in next_gw_fixtures:
    home_team_id = fixture['team_h']
    away_team_id = fixture['team_a']
    home_team_name = team_id_to_name.get(home_team_id, "Unknown Team")
```

```
    away_team_name = team_id_to_name.get(away_team_id, "Unknown Team")
    home_position = team_position.get(home_team_id, "Unknown Position")
    away_position = team_position.get(away_team_id, "Unknown Position")
    if abs(int(home_position) - int(away_position)) >= 5:
        if home_position > away_position:
            Underdog_Bonus = 'Home'
        else:
            Underdog_Bonus = 'Away'
    else:
        Underdog_Bonus = 'None'

    wait = WebDriverWait(driver, 10)
    try:
        close_ad = wait.until(EC.element_to_be_clickable((By.CLASS_NAME,␣
↪'webpush-swal2-close')))
        close_ad.click()
    except TimeoutException:
        print()

    home_team = TEAM_NAMES_ODDSCHECKER.get(home_team_name, home_team_name)
    away_team = TEAM_NAMES_ODDSCHECKER.get(away_team_name, away_team_name)
    match_title = home_team + " v " + away_team

    matches_button = driver.find_element(By.XPATH, "//button[contains(text(),␣
↪'Matches')]")
    matches_button.click()

    # Find match link
    match_link = driver.find_element(By.XPATH, f"//
↪a[@title='{match_title}'][@href]")
    href = match_link.get_attribute("href")
    driver.get(href)

    try:
        win_market_header = driver.find_element(By.XPATH, "//
↪h2[contains(text(), 'Win Market')]")
        # Expand the section if it's collapsed
        if win_market_header.get_attribute("aria-expanded") == "false":
            win_market_header.click()
            time.sleep(3)
        outcomes = driver.find_elements(By.XPATH, "//h2[contains(text(), 'Win␣
↪Market')]/following-sibling::*[1]/*[1]/*[1]//p")
        for outcome in outcomes:
            odd = outcome.find_element(By.XPATH, "./following-sibling::button")
            if outcome.get_attribute("innerText") == home_team:
                home_win = odd.get_attribute("innerText")
            elif outcome.get_attribute("innerText") == away_team:
```

```
                away_win = odd.get_attribute("innerText")
            else:
                draw = odd.get_attribute("innerText")

    except Exception as e:
        print()

    try:
        # Find the "Total Home Goals" market
        total_home_goals_header = driver.find_element(By.XPATH, "//h2[text() =␣
↪'Total Home Goals']")
        total_away_goals_header = driver.find_element(By.XPATH, "//h2[text() =␣
↪'Total Away Goals']")
        # Expand the section if it's collapsed
        if total_home_goals_header.get_attribute("aria-expanded") == "false":
            total_home_goals_header.click()
            time.sleep(3)

        if total_away_goals_header.get_attribute("aria-expanded") == "false":
            total_away_goals_header.click()
            time.sleep(3)
    except Exception as e:
        print()

    try:
        over_05_row_home = total_home_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '0.5')]/
↪following-sibling::button")
    except NoSuchElementException:
        total_home_goals_header.click()
        time.sleep(3)
        total_home_goals_header.click()

    try:
        over_05_row_home = total_home_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '0.5')]/
↪following-sibling::button")
        away_to_concede_odds = over_05_row_home.get_attribute("innerText")
        home_0_goal = 1 - (1/(float(Fraction(away_to_concede_odds)) + 1))
    except NoSuchElementException:
        home_0_goal = 0
    try:
        over_15_row_home = total_home_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '1.5')]/
↪following-sibling::button")
        home_over_15 = over_15_row_home.get_attribute("innerText")
```

```
        home_1_goal = (1 - (1/(float(Fraction(home_over_15)) + 1))) -␣
↪home_0_goal
        home_2_goal = (1/(float(Fraction(home_over_15)) + 1))
    except NoSuchElementException:
        home_1_goal = 0
        home_2_goal = 0

    try:
        over_25_row_home = total_home_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '2.5')]/
↪following-sibling::button")
        home_over_25 = over_25_row_home.get_attribute("innerText")
        home_2_goal = (1 - (1/(float(Fraction(home_over_25)) + 1))) -␣
↪home_1_goal - home_0_goal
        home_3_goal = (1/(float(Fraction(home_over_25)) + 1))
    except NoSuchElementException:
        home_3_goal = 0

    try:
        over_35_row_home = total_home_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '3.5')]/
↪following-sibling::button")
        home_over_35 = over_35_row_home.get_attribute("innerText")
        home_3_goal = (1 - (1/(float(Fraction(home_over_35)) + 1))) -␣
↪home_2_goal - home_1_goal - home_0_goal
        home_4_goal = (1/(float(Fraction(home_over_35)) + 1))
    except NoSuchElementException:
        try:
            under_35_row_home = total_home_goals_header.find_element(By.XPATH,␣
↪"./following-sibling::*[1]/*[1]/*[3]/*[1]//p[contains(text(), '3.5')]/
↪following-sibling::button")
            home_under_35 = under_35_row_home.get_attribute("innerText")
            home_3_goal = (1/(float(Fraction(home_under_35)) + 1)) -␣
↪home_2_goal - home_1_goal - home_0_goal
            home_4_goal = 1 - (1/(float(Fraction(home_under_35)) + 1))
        except NoSuchElementException:
            try:
                under_45_row_home = total_home_goals_header.find_element(By.
↪XPATH, "./following-sibling::*[1]/*[1]/*[3]/*[1]//p[contains(text(), '4.5')]/
↪following-sibling::button")
                home_under_45 = under_45_row_home.get_attribute("innerText")
                home_4_goal = (1/(float(Fraction(home_under_45)) + 1)) -␣
↪max(home_3_goal, 0) - home_2_goal - home_1_goal - home_0_goal
            except NoSuchElementException:
                home_4_goal = 0
```

```
    try:
        over_05_row_away = total_away_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '0.5')]/
↪following-sibling::button")
    except NoSuchElementException or ElementClickInterceptedException:
        home_4_goal = 0
        time.sleep(3)
        total_away_goals_header.click()

    try:
        over_05_row_away = total_away_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '0.5')]/
↪following-sibling::button")
        home_to_concede_odds = over_05_row_away.get_attribute("innerText")
        away_0_goal = 1 - (1/(float(Fraction(home_to_concede_odds)) + 1))
    except NoSuchElementException:
        away_0_goal = 0

    try:
        over_15_row_away = total_away_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '1.5')]/
↪following-sibling::button")
        away_over_15 = over_15_row_away.get_attribute("innerText")
        away_1_goal = (1 - (1/(float(Fraction(away_over_15)) + 1))) -␣
↪away_0_goal
        away_2_goal = (1/(float(Fraction(away_over_15)) + 1))
    except NoSuchElementException:
        away_1_goal = 0
        away_2_goal = 0
    try:
        over_25_row_away = total_away_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '2.5')]/
↪following-sibling::button")
        away_over_25 = over_25_row_away.get_attribute("innerText")
        away_2_goal = (1 - (1/(float(Fraction(away_over_25)) + 1))) -␣
↪away_1_goal - away_0_goal
        away_3_goal = (1/(float(Fraction(away_over_25)) + 1))
    except NoSuchElementException:
        away_3_goal = 0
    try:
        over_35_row_away = total_away_goals_header.find_element(By.XPATH, "./
↪following-sibling::*[1]/*[1]/*[3]/*[2]//p[contains(text(), '3.5')]/
↪following-sibling::button")
        away_over_35 = over_35_row_away.get_attribute("innerText")
        away_3_goal = (1 - (1/(float(Fraction(away_over_35)) + 1))) -␣
↪away_2_goal - away_1_goal - away_0_goal
```

```
        away_4_goal = (1/(float(Fraction(away_over_35)) + 1))

    except NoSuchElementException:
        try:
            under_35_row_away = total_away_goals_header.find_element(By.XPATH,␣
↪"./following-sibling::*[1]/*[1]/*[3]/*[1]//p[contains(text(), '3.5')]/
↪following-sibling::button")
            away_under_35 = under_35_row_away.get_attribute("innerText")
            away_3_goal = (1/(float(Fraction(away_under_35)) + 1)) -␣
↪away_2_goal - away_1_goal - away_0_goal
            away_4_goal = 1 - (1/(float(Fraction(away_under_35)) + 1))
        except NoSuchElementException:
            try:
                under_45_row_away = total_away_goals_header.find_element(By.
↪XPATH, "./following-sibling::*[1]/*[1]/*[3]/*[1]//p[contains(text(), '4.5')]/
↪following-sibling::button")
                away_under_45 = under_45_row_away.get_attribute("innerText")
                away_4_goal = (1/(float(Fraction(away_under_45)) + 1)) -␣
↪max(away_3_goal, 0) - away_2_goal - away_1_goal - away_0_goal
            except NoSuchElementException:
                away_4_goal = 0

    Manager_Points_Home = (1/(float(Fraction(home_win)) + 1)) * 6 + (1/
↪(float(Fraction(draw)) + 1)) * 3 + away_0_goal * 2 + home_1_goal + 2 *␣
↪home_2_goal + 3 * home_3_goal + 4 * max(home_4_goal, 0)
    Manager_Points_Away = (1/(float(Fraction(away_win)) + 1)) * 6 + (1/
↪(float(Fraction(draw)) + 1)) * 3 + home_0_goal * 2 + away_1_goal + 2 *␣
↪away_2_goal + 3 * away_3_goal + 4 * max(away_4_goal, 0)
    if Underdog_Bonus == 'Home':
        Manager_Points_Home += ((1/(float(Fraction(home_win)) + 1)) * 10 + (1/
↪(float(Fraction(draw)) + 1)) * 5)
    if Underdog_Bonus == 'Away':
        Manager_Points_Away += ((1/(float(Fraction(away_win)) + 1)) * 10 + (1/
↪(float(Fraction(draw)) + 1)) * 5)

    data.append({
            'Home Team': home_team_name,
            'Away Team': away_team_name,
            'Home Win Odds': (1/(float(Fraction(home_win)) + 1)),
            'Draw Odds': (1/(float(Fraction(draw)) + 1)),
            'Away Win Odds': (1/(float(Fraction(away_win)) + 1)),
            'Home Clean Sheet %': away_0_goal*100,
            'Away Clean Sheet %': home_0_goal*100,
            'Home to Score 1 Goal': home_1_goal,
            'Home to Score 2 Goals': home_2_goal,
            'Home to Score 3 Goals': home_3_goal,
```

```
            'Home to Score 4 Goals': max(home_4_goal, 0),
            'Away to Score 1 Goal': away_1_goal,
            'Away to Score 2 Goals': away_2_goal,
            'Away to Score 3 Goals': away_3_goal,
            'Away to Score 4 Goals': max(away_4_goal, 0),
            'Manager Bonus For': Underdog_Bonus,
            'Home Manager Points': Manager_Points_Home,
            'Away Manager Points': Manager_Points_Away

        })
    driver.get("https://www.oddschecker.com/football/english/premier-league/")

driver.quit()
```

[15]:
```
df = pd.DataFrame(data)
print(df.head())
df.to_excel("output.xlsx", index=False)
```

```
        Home Team    Away Team  Home Win Odds  Draw Odds  Away Win Odds  \
0   Nott'm Forest     Man City       0.266667   0.263158       0.505051
1        Brighton       Fulham       0.476190   0.277778       0.277778
2  Crystal Palace      Ipswich       0.666667   0.217391       0.142857
3       Liverpool  Southampton       0.888889   0.083333       0.043478
4       Brentford  Aston Villa       0.400000   0.263158       0.354839

   Home Clean Sheet %  Away Clean Sheet %  Home to Score 1 Goal  \
0           13.333333           25.000000              0.350000
1           26.666667           16.666667              0.320513
2           40.000000           10.000000              0.263636
3           50.000000            1.492537              0.102722
4           18.181818           15.384615              0.322344

   Home to Score 2 Goals  Home to Score 3 Goals  Home to Score 4 Goals  \
0               0.233333               0.142276               0.024390
1               0.262821               0.191176               0.058824
2               0.279221               0.232143               0.125000
3               0.190045               0.692308               0.000000
```

```
4               0.260652               0.196491               0.066667

   Away to Score 1 Goal  Away to Score 2 Goals  Away to Score 3 Goals  \
0              0.295238               0.277311               0.217195
1              0.369697               0.220779               0.123249
2              0.377778               0.155556               0.048810
3              0.318182               0.129187               0.038547
4              0.330377               0.257036               0.183150

   Away to Score 4 Goals Manager Bonus For  Home Manager Points  \
0               0.076923              None             3.997197
1               0.019608              None             5.878787
2               0.017857              Away             7.470680
3               0.014085              Away             9.143069
4               0.047619              None             5.252898

   Away Manager Points
0             6.128913
1             4.092768
2             5.131591
3             2.140704
4             4.810573
```