Pynutlib Use-case Scenario

Travis Ke NCCU

May 25, 2025

Section 2 - Pynutlib

We **published a package with PyPi.org** which business can immediately use as a business analytics tool to assess their items on their menu.

Here is one use-case scenario for demo, the business owner is Wendy's

The Business' Data Analysis Group Can Access our Product with these 2 line of codes

[4]:
```
! pip install pynutlib # version 0.1.1
from pynut import API_KEY, calculate_tee, score_menu,␣
 ↪compute_target_macros_per_meal
```

```
Requirement already satisfied: pynutlib in /opt/anaconda3/lib/python3.12/site-
packages (0.1.0)
Requirement already satisfied: streamlit in /opt/anaconda3/lib/python3.12/site-
packages (from pynutlib) (1.37.1)
Requirement already satisfied: requests in /opt/anaconda3/lib/python3.12/site-
packages (from pynutlib) (2.32.3)
Requirement already satisfied: pandas in /opt/anaconda3/lib/python3.12/site-
packages (from pynutlib) (2.2.2)
Requirement already satisfied: numpy in /opt/anaconda3/lib/python3.12/site-
packages (from pynutlib) (1.26.4)
Requirement already satisfied: matplotlib in /opt/anaconda3/lib/python3.12/site-
packages (from pynutlib) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (24.1)
Requirement already satisfied: pillow>=8 in /opt/anaconda3/lib/python3.12/site-
packages (from matplotlib->pynutlib) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
```

```
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in
/opt/anaconda3/lib/python3.12/site-packages (from matplotlib->pynutlib)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
/opt/anaconda3/lib/python3.12/site-packages (from pandas->pynutlib) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
/opt/anaconda3/lib/python3.12/site-packages (from pandas->pynutlib) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->pynutlib) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->pynutlib) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->pynutlib) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->pynutlib)
(2025.4.26)
Requirement already satisfied: altair<6,>=4.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (5.0.1)
Requirement already satisfied: blinker<2,>=1.0.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (1.6.2)
Requirement already satisfied: cachetools<6,>=4.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (5.3.3)
Requirement already satisfied: click<9,>=7.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (8.1.7)
Requirement already satisfied: protobuf<6,>=3.20 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (4.25.3)
Requirement already satisfied: pyarrow>=7.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (16.1.0)
Requirement already satisfied: rich<14,>=10.14.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (13.7.1)
Requirement already satisfied: tenacity<9,>=8.1.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (8.2.3)
Requirement already satisfied: toml<2,>=0.10.1 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (0.10.2)
Requirement already satisfied: typing-extensions<5,>=4.3.0 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (4.11.0)
Requirement already satisfied: gitpython!=3.1.19,<4,>=3.0.7 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (3.1.43)
Requirement already satisfied: pydeck<1,>=0.8.0b4 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (0.8.0)
Requirement already satisfied: tornado<7,>=6.0.3 in
/opt/anaconda3/lib/python3.12/site-packages (from streamlit->pynutlib) (6.4.1)
Requirement already satisfied: jinja2 in /opt/anaconda3/lib/python3.12/site-
packages (from altair<6,>=4.0->streamlit->pynutlib) (3.1.4)
Requirement already satisfied: jsonschema>=3.0 in
/opt/anaconda3/lib/python3.12/site-packages (from
altair<6,>=4.0->streamlit->pynutlib) (4.23.0)
```

```
Requirement already satisfied: toolz in /opt/anaconda3/lib/python3.12/site-
packages (from altair<6,>=4.0->streamlit->pynutlib) (0.12.0)
Requirement already satisfied: gitdb<5,>=4.0.1 in
/opt/anaconda3/lib/python3.12/site-packages (from
gitpython!=3.1.19,<4,>=3.0.7->streamlit->pynutlib) (4.0.7)
Requirement already satisfied: six>=1.5 in /opt/anaconda3/lib/python3.12/site-
packages (from python-dateutil>=2.7->matplotlib->pynutlib) (1.16.0)
Requirement already satisfied: markdown-it-py>=2.2.0 in
/opt/anaconda3/lib/python3.12/site-packages (from
rich<14,>=10.14.0->streamlit->pynutlib) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
/opt/anaconda3/lib/python3.12/site-packages (from
rich<14,>=10.14.0->streamlit->pynutlib) (2.15.1)
Requirement already satisfied: smmap<5,>=3.0.1 in
/opt/anaconda3/lib/python3.12/site-packages (from
gitdb<5,>=4.0.1->gitpython!=3.1.19,<4,>=3.0.7->streamlit->pynutlib) (4.0.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/opt/anaconda3/lib/python3.12/site-packages (from
jinja2->altair<6,>=4.0->streamlit->pynutlib) (2.1.3)
Requirement already satisfied: attrs>=22.2.0 in
/opt/anaconda3/lib/python3.12/site-packages (from
jsonschema>=3.0->altair<6,>=4.0->streamlit->pynutlib) (23.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in
/opt/anaconda3/lib/python3.12/site-packages (from
jsonschema>=3.0->altair<6,>=4.0->streamlit->pynutlib) (2023.7.1)
Requirement already satisfied: referencing>=0.28.4 in
/opt/anaconda3/lib/python3.12/site-packages (from
jsonschema>=3.0->altair<6,>=4.0->streamlit->pynutlib) (0.30.2)
Requirement already satisfied: rpds-py>=0.7.1 in
/opt/anaconda3/lib/python3.12/site-packages (from
jsonschema>=3.0->altair<6,>=4.0->streamlit->pynutlib) (0.10.6)
Requirement already satisfied: mdurl~=0.1 in /opt/anaconda3/lib/python3.12/site-
packages (from markdown-it-py>=2.2.0->rich<14,>=10.14.0->streamlit->pynutlib)
(0.1.0)
```

The Business' Data Analysis Group Can Access our Product with these 2 line of codes

[5]:
```
# Simulated customer database (we don't have access to Wendy's customer database
# , but even if they don't, they can just do a survey to acquire the data needed.
 ↪)

customers = [
    {'gender': 'male', 'age': 23, 'height': 180, 'weight': 70, 'activity_level':␣
 ↪'inactive'},
    {'gender': 'female', 'age': 30, 'height': 165, 'weight': 60,␣
 ↪'activity_level': 'active'},
```

```
    {'gender': 'male', 'age': 40, 'height': 175, 'weight': 80, 'activity_level':␣
 ↪'low active'},
    {'gender': 'female', 'age': 28, 'height': 160, 'weight': 55,␣
 ↪'activity_level': 'very active'}
]

# List to store calculated TEEs
tee_values = []

# Loop through each customer and calculate TEE
for customer in customers:
    tee = calculate_tee(
        customer['gender'],
        customer['age'],
        customer['height'],
        customer['weight'],
        customer['activity_level']
    )
    tee_values.append(tee)

# Calculate average TEE
average_tee = sum(tee_values) / len(tee_values)

# Print result
print(f"Average TEE: {average_tee:.2f} kcal/day")
```

```
Average TEE: 2558.49 kcal/day
```

Web-scrape Restaurant's Menu - Wendy's

[11]:
```
import pandas as pd                    # Import pandas for handling and␣
 ↪manipulating tabular data (e.g., DataFrames)
from bs4 import BeautifulSoup          # Import BeautifulSoup for parsing HTML/
 ↪XML content (not used in this notebook yet)
import requests
```

[7]:
```
# Correct full URL of Wendy's nutrition page
url = 'https://www.wendys.co.nz/our-food/nutritional'

# Add verify=False to skip SSL certificate verification during testing
response = requests.get(url, verify=False)

if response.status_code == 200: # This line checks whether the HTTP request␣
 ↪succeeded.(The code, 200, is the standard HTTP status code for OK)
    soup = BeautifulSoup(response.text, 'html.parser')

    table = soup.find('tbody')  # Locate the main table body
```

```
    rows = table.find_all('tr')  # Extract all rows from the table

    items = []  # Store parsed nutritional data

    for row in rows:
        cols = row.find_all('td')

        # Ensure the row contains at least 11 data cells (actual data row)
        if len(cols) >= 10:
            try:
                item_name = cols[0].text.strip()
                serving_size = cols[1].text.strip()
                weight = float(cols[2].text.strip())
                calories = float(cols[4].text.strip())
                protein = float(cols[5].text.strip())
                total_fat = float(cols[6].text.strip())
                saturated_fat = float(cols[7].text.strip())
                carbohydrates = float(cols[8].text.strip())
                sugars = float(cols[9].text.strip())
                sodium = float(cols[10].text.strip())

            except ValueError:
                # Skip rows with invalid numeric values (e.g., empty or␣
↪non-convertible)
                continue

            # Add parsed row to list as a structured dictionary
            items.append({
                'Item': item_name,
                'Serving Size': serving_size,
                'Weight (g)': weight,
                'Calories': calories,
                'Protein (g)': protein,
                'Total Fat (g)': total_fat,
                'Saturated Fat (g)': saturated_fat,
                'Carbohydrates (g)': carbohydrates,
                'Sugars (g)': sugars,
                'Sodium (mg)': sodium
            })

    # Convert the list of dictionaries to a pandas DataFrame
    df = pd.DataFrame(items)

    print("Successfully parsed nutritional data!")
else:
    print(" Failed to fetch data, status code: ", response.status_code)
```

```
/opt/anaconda3/lib/python3.12/site-packages/urllib3/connectionpool.py:1099:
```

```
InsecureRequestWarning: Unverified HTTPS request is being made to host
'www.wendys.co.nz'. Adding certificate verification is strongly advised. See:
https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(

Successfully parsed nutritional data!
```

[8]:
```
df
```

[8]:

	Item	Serving Size	Weight (g)	Calories	Protein (g)	Total Fat (g)	Saturated Fat (g)	Carbohydrates (g)	Sugars (g)	Sodium (mg)
0	1/4 lb Single	1 ea.	277.0	604.0	26.0	31.0	16.0	36.0	11.0	731.0
1	1/2 lb Double	1 ea.	351.0	985.0	50.0	61.0	33.0	36.0	11.0	859.0
2	3/4 lb Triple	1 ea.	435.0	1367.0	72.0	90.0	50.0	36.0	11.0	987.0
3	Big Bacon Classic	1 ea.	295.8	707.0	34.0	40.0	21.0	36.0	11.1	980.0
4	Baconator	1 ea.	312.0	1061.0	57.0	68.0	36.0	34.0	9.0	1143.0
..	...	...	...	...	...	...	...	...	...	...
111	Calci-Yum Chocolate	1 pk.	250.0	150.5	8.5	3.5	2.0	20.5	20.0	105.0
112	Calci-Yum Strawberry	1 pk.	250.0	147.5	8.0	3.3	2.0	21.0	21.0	102.5
113	Calci-Yum Banana	1 pk.	250.0	148.7	8.0	3.3	2.0	21.3	21.3	102.5
114	Strawberry Daiquiri Sparkler	20 oz	591.0	363.2	0.0	0.0	0.0	88.7	88.7	90.5
115	Cherry cola	20 oz	591.0	494.1	0.1	0.0	0.0	114.3	114.2	60.6

```
[116 rows x 10 columns]
```

[9]:
```
goal=compute_target_macros_per_meal(average_tee)
goal
```

[9]:
```
{'Protein (g)': 85.28291666666668,
 'Fat (g)': 28.42763888888889,
 'Carbs (g)': 63.9621875}
```

[10]:
```
# Rename necessary columns
df = df.rename(columns={
    "Total Fat (g)": "Fat (g)",
    "Carbohydrates (g)": "Carbs (g)"
})
score_menu(df, goal, tee, "muscle_gain")
```

[10]:

	Item	Serving Size	Weight (g)	\
27	Chickenator	1ea	374.4	
2	3/4 lb Triple	1 ea.	435.0	
21	Chicken Club	1 ea.	283.8	
22	Avocado Bacon Supreme	1 ea.	307.2	
19	Homestyle Chicken Burger	1 ea.	245.0	
..	...	...	...	
105	Coke Zero	20oz	591.0	
42	Large lettuce leaf for low carb burger	1 leaf	24.0	
45	Red Onions	2 rings	7.0	
41	Lettuce	1 leaf	15.0	
46	Dill Pickles	3 ea.	8.0	

	Calories	Protein (g)	Fat (g)	Saturated Fat (g)	Carbs (g)	Sugars (g)	\
27	959.0	58.0	43.0	9.0	75.0	9.0	
2	1367.0	72.0	90.0	50.0	36.0	11.0	
21	616.0	34.0	26.0	6.0	53.0	8.0	
22	626.0	32.0	25.0	4.0	57.0	12.0	
19	537.0	28.0	19.0	2.0	54.0	9.0	
..	...	...	...	...	...	...	
105	2.0	0.3	0.0	0.0	0.6	0.0	
42	0.8	0.4	0.1	0.0	0.2	0.0	
45	1.7	0.1	0.0	0.0	0.3	0.3	
41	0.3	0.2	0.0	0.0	0.1	0.0	
46	0.0	0.0	0.0	0.0	0.0	0.0	

	Sodium (mg)	Calories Score	Protein Score	Fat Score	Carbs Score	\
27	2872.0	0.390226	0.680089	0.487388	0.827432	
2	987.0	0.556245	0.844249	0.000000	0.562833	
21	1641.0	0.250656	0.398673	0.914603	0.828615	
22	1678.0	0.254725	0.375222	0.879426	0.891152	

```
19        1344.0        0.218510        0.328319   0.668364     0.844249
..           ...             ...             ...        ...          ...
105         86.0        0.000814        0.003518   0.000000     0.009381
42           5.4        0.000326        0.004690   0.003518     0.003127
45           0.9        0.000692        0.001173   0.000000     0.004690
41           2.1        0.000122        0.002345   0.000000     0.001563
46         109.1        0.000000        0.000000   0.000000     0.000000

     Total Score
27      0.613045
2       0.561515
21      0.558244
22      0.555149
19      0.477552
..           ...
105     0.003446
42      0.003270
45      0.001545
41      0.001275
46      0.000000

[116 rows x 15 columns]
```

Possible Application: Wendy's Use Case

Wendy's can leverage our **pynutlib** package in several ways:

1. **Smart Menu Optimization**
   Wendy's can use our scoring functions to create a **composite nutrition score** — for example, combining *protein per fat ratio*, *caloric density*, or *macro balance*. This allows them to:
   - Design custom **meal bundles** for specific customer goals (e.g., bulking, weight loss, or high protein diets).
   - Highlight top-ranked items to promote healthier eating habits.
2. **App Integration**
   By integrating our package into their backend or mobile app, Wendy's can:
   - Provide **personalized food suggestions** based on user goals (e.g., "Show me meals under 500 kcal and high in protein").
   - Let customers **filter or sort menu items** based on our score system — turning static menus into dynamic, goal-based experiences.
3. **Business Intelligence Tool**
   Their internal analytics team can:
   - Import the Pynutlib package using just two lines of code.
   - Analyze nutritional profiles across all menu items instantly.
   - Monitor **customer TEE averages** to plan menu adjustments seasonally or demographically.

[ ]: